Australian Government
Department of Defence
Defence Science and
Technology Organisation

Detection and Classification of Objects in Synthetic Aperture Radar Imagery

Tristrom Cooke

Intelligence, Surveillance and Reconnaissance Division
Defence Science and Technology Organisation

DSTO–RR–0305

ABSTRACT

This is an amalgamation of a number of reports written by the author when he was contracted to DSTO through CSSIP. The reports concern the detection of faint trails, and the theory and evaluation of a number of existing and novel methods for the detection and classification of ground and maritime targets within SAR imagery. The publication of this collection allows the results to be available within defence and the wider community.

Published by

Defence Science and Technology Organisation
PO Box 1500
Edinburgh, South Australia 5111, Australia

Telephone: (08) 8259 5555
Facsimile: (08) 8259 6567

AR No. 013-591
February 2006

APPROVED FOR PUBLIC RELEASE

Detection and Classification of Objects in Synthetic Aperture Radar Imagery

EXECUTIVE SUMMARY

The original concept of ADSS (Analysts' Detection Support System) was to assist analysts with the difficult and onerous task of detecting vehicles in SAR (Synthetic Aperture Radar) imagery. It was realised however that many of the detection and classification algorithms would have wider application than just this one task, and so ADSS has expanded its capabilities to also process ISAR (Inverse Synthetic Aperture Radar), hyper-spectral, geo-spatial and video data. Because of this potential for general application, a number of CSSIP (CRC for Sensor Signals and Information Processing) reports written by the author while contracted to DSTO, have been collected in this document so that their contents would be available to defence and the wider community.

The first of the collected reports deals with the automatic detection of faint trails within images. This is done with a novel multi-scale method based on the radial derivative of the Radon transform of the image, which is outlined in the report. The planned use of this algorithm was to fuse this information with any detected ground targets within the imagery to assess the threat level of that target. It may also prove useful for automatic registration between two images.

The next group of three reports concerns the extraction of features from SAR images of ground-targets. This builds upon similar work conducted at Lincoln Labs, which found a "best" set of features on which to distinguish targets from background clutter for their radar system. These reports describe the evaluation of these features (as well as a number of other novel features and useful features from related pattern recognition tasks such as texture matching) for imagery obtained from the INGARA radar platform.

Following these is a brief summary report on some existing classification methods, with some additional new analysis on linear discriminants.

The final two reports focus on maritime detection. Like the previous detection tasks, this consists of prescreening, Low Level Classification (LLC) and possibly a high level classification stage.

The first of the maritime detection reports focusses on prescreening. The standard prescreener is ATA (Adaptive Threshold Algorithm), and this has been extensively tested on various data, and empirically compared against other parametric and non-parametric histogram based prescreeners. Surprisingly, for the particular data tested, the detector which most accurately modelled the background distribution (the newly described Hill's detector) gave the worst performance of those tested. The template detector, which gave the best performance, was not much better than simple ATA.

The last report investigates a number of algorithms which may be useful alternatives to the LLC modules currently implemented in ADSS. Two categories of LLC methods were considered. The first was feature extraction, where a number of rotation and translation invariant features were described and tested. The second was classification, which in this report focussed on work relating to ensemble classifiers, which produce a well performing

classifier by combining a large number of simpler base classifiers. As well as the standard boosting and bagging methods described in the literature, some novel methods for combining classifiers were also introduced, and tested on a simple data set.

Author

Tristrom Cooke
CRC for Sensor Signals and Information Processing

The author obtained a B.Eng in Electrical Engineering from the University of South Australia in 1992, a B.Sc (Hons) in Applied Mathematics from the University of Adelaide in 1995, and a PhD in Applied Mathematics (also at the University of Adelaide) in the area of thermoelasticity in 1999. Until 2005, he was employed by CSSIP (CRC for Sensor Signals and Information Processing) where this report was written, but is now employed by DSTO in ISRD. He has mostly worked on projects relating to recognition of targets in both SAR and ISAR radar imagery.

Contents

Figures

Preface

This document is a lightly edited collection of CSSIP reports commissioned by DSTO and written by the author. These reports were not originally accessible, and they have been reissued here as a DSTO report to make them referable and accessible to a wider audience. Each chapter is a separate report, the details of which are as follows:

- Chapter 1 was originally "Report on faint trail detection," CSSIP CR-4/99, February 1999.
- Chapter 2 was originally "First report on features for target/background classification," CSSIP CR-9/99, April 1999.
- Chapter 3 was originally "Second report on features for target/background classification," CSSIP CR-26/99, November 1999.
- Chapter 4 was originally "Third report on features for target/background classification," CSSIP CR-5/00, May 2000.
- Chapter 5 was originally "Discriminant based classification," CSSIP CR-25/99, October 1999.
- Chapter 6 was originally "SAR image analysis in maritime contexts: Prescreening," CSSIP CR-20/03, December 2003.
- Chapter 7 was originally titled "SAR image analysis in maritime contexts: Low level Classification," and was handed to DSTO in 2005, although it was not a formal deliverable and has no CSSIP report number.

Some of the material from these reports was also used to write the following external publications:

- T.Cooke, "A Radon transform derivative method for faint trail detection in SAR imagery." In DICTA'99 conference proceedings, pp.31-34, 1999.
- T.Cooke, N.J.Redding, J.Schroeder and J.Zhang, "Comparison of selected features for target detection in synthetic aperture radar imagery," Digital Signal Processing, Vol.10, No.4, pp 286-296, October 2000.
- T.Cooke, "Two variations on Fisher's linear discriminant for pattern recognition," IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.24, No.2, pp 268-273, February 2002.

- T.Cooke and M.Peake, "The optimal classification using a linear discriminant for two point classes having known mean and covariance," Journal of Multivariate Analysis, Vol.82, No.2, pp 379-394, August 2002.

Chapter 1

Faint Trail Detection

1.1 Introduction

There has been a large amount of interest in the area of computational line detection. The earliest line detectors relied on the detection of edges over small and localised areas. Square masks of size 2×2 (for the Roberts cross-gradient method) or 3×3 (for the Prewitt or Sobel operators) are translated over the image, and the magnitude of the gradient is estimated at each point. Positions in the image having a gradient above a certain threshold are assumed to correspond to an edge. One problem in the application of this method to Synthetic Aperture Radar (SAR) imagery is the significant amount of speckle noise present, which results in a higher False Alarm Rate (FAR).

Currently, a standard edge detector is the Canny algorithm, which is available with the mathematics package MATLAB. This method similarly fails to work in SAR imagery due to both the presence of noise, and the variations in the natural background which the algorithm is unable to distinguish from a road.

Most other types of feature detectors rely implicitly on some knowledge of the type of objects they are searching for, which would not make them suitable for the current application, since the shape of the features searched for are not known a priori. For instance, methods which find peaks in either the Radon or the Hough transform of the entire image assume that the object to be located can be accurately represented by a perfectly straight line or a quadratic polynomial respectively, which is usually not the case.

The standard technique which has met with the best success so far has been Steger's method [3], which approximates the road profile by a second order polynomial. The zero crossings of the second derivative are marked as edges. This method was found to work extremely well for main roads, and was subsequently incorporated into code for map to image registration. The fainter roads which are easily detectable by eye were still unable to be detected well with Steger's method, so some other methods were devised. At the time of the Mid First Year Six Month Report, November 1998, the most promising candidate was Garry Newsam's hypothesis testing algorithm [2].

A more detailed explanation of this algorithm is presented in section 1.3.1, but the main idea behind the method is to examine line segments within the image, and deter-

mine if the intensity distribution of the pixels on these segments are statistically significant from the background distribution. If the test concluded that there were statistically significant differences, then that line was presumed to be a part of a road. Two slightly differing algorithms had been suggested for the implementation of this procedure. The first, and simpler case, was the uncorrelated case where the grey levels of neighbouring pixels were assumed to be completely unrelated to each other. This method produced results favourable to those produced by Steger's method in that it detected main roads well, but often failed to detect the fainter roads. The level of false alarms produced was also quite high, since it also detected small hills and other statistically anomalous objects. The second case assumed a particular form of correlation between the pixels of the image.

At the time of the last six monthly report, some code for this uncorrelated hypothesis test had been written, but was not yet fully working. It was expected that this correlated case would produce much better results for faint trails. Since that time, the code for this has been completed, and it has been found that the results of this more accurate model in fact works less well than the uncorrelated procedure. Some possible explanations for this are given in section 1.3.1.

After another false start (based on the Karhunen Loeve transform), a technique using the radial derivative of the Radon transform (RDRT) was developed, and found to detect both main and faint roads with a high PD (Probability of Detection) and a low FAR. Although it has limited success in detecting natural curvilinear features such as small creeks (which other detectors fail to detect entirely), it is very suitable for the detection of faint lines in SAR images.

1.2 Background

This section describes the 'sliding window' technique for calculating block Radon transforms, which is used in both the hypothesis testing and RDRT algorithms.

The Radon transform of an image is a function of two parameters ρ and θ which corresponds to the integral over the line whose point closest to the origin can be written in polar coordinates as (ρ, θ). If the Radon transform is calculated over a rectangular block, then the lines near to the corner of the block will be much shorter than those closer to the center, and so will not contain as much information and hence will be less statistically significant. The way used in this report to avoid these smaller lines, yet still consider all of the line segments within the image, is to consider the sliding window arrangement shown in Figure 1.1. The Radon transforms are calculated in the overlapping subimages, and then the transform domain is restricted to the inner windows or Radon blocks, which are non-overlapping.

Even when restricted to the inner window, there will be some variation in the value of the Radon transform due to changes in the length of the lines over which the pixel intensities are integrated. These variations are removed by dividing each Radon transform by the length of the line in the subimage. These lengths may easily be calculated by simply calculating the Radon transform of a rectangle of the same size as the subimage, but containing all ones. Although the resulting function of ρ and θ is no longer strictly a Radon transform, this is how it shall be referred for the remainder of this report.

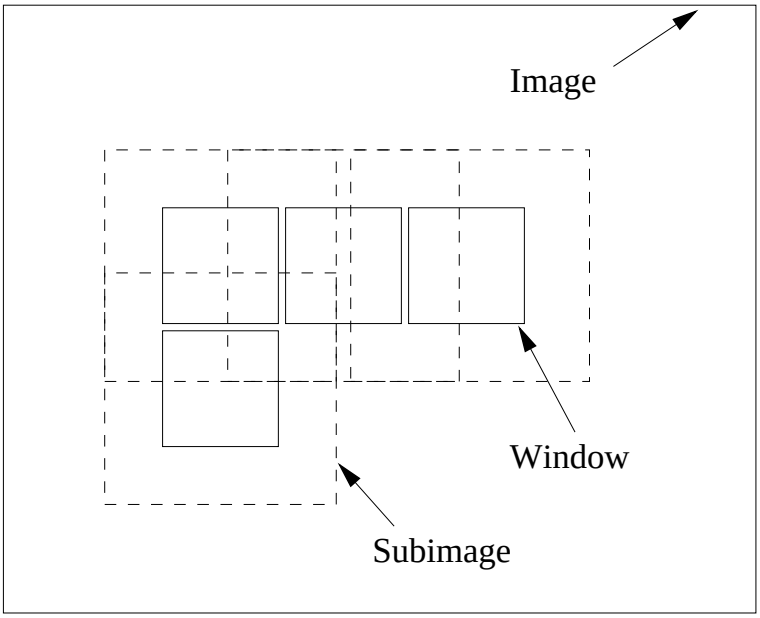


Figure 1.1: *Sliding window arrangement in an image*

1.3 Some Algorithms for Road Detection

1.3.1 Hypothesis Testing - Correlated Case

In the previous report, a special case of Garry Newsam's hypothesis testing algorithm [2] was presented. For this case, it was assumed that there was no degree of correlation between the gray levels of neighbouring pixels. Since that time, the related correlated case has been examined.

Both hypothesis testing algorithms work by examining two hypothesis. The first is that all of the pixels on a particular line is produced by the same statistical distribution as the background. The second hypothesis is that the statistics of the points on the line are generated by a completely different distribution from that of the background. The probability likelihoods that each of these hypotheses are true is calculated, and the ratio of these probabilities (the maximum likelihood ratio) is produced. If this ratio is below a certain threshold, then it is much more likely that the pixel intensity values on the line was generated by a distribution that is different from background, and so it is marked as a road. The only difference between the two algorithms is that the first assumes that there is no correlation between neighbouring pixels, while the second assumes that there is an exponential correlation between successive pixels (*i.e.* $c_n = c_0(b)^{|n|}$, where c_n is the correlation between pixels separated by a distance n and $b = c_1/c_0$).

For the correlated case, Garry Newsam [2] gives the logarithm of the maximum likelihood ratio as

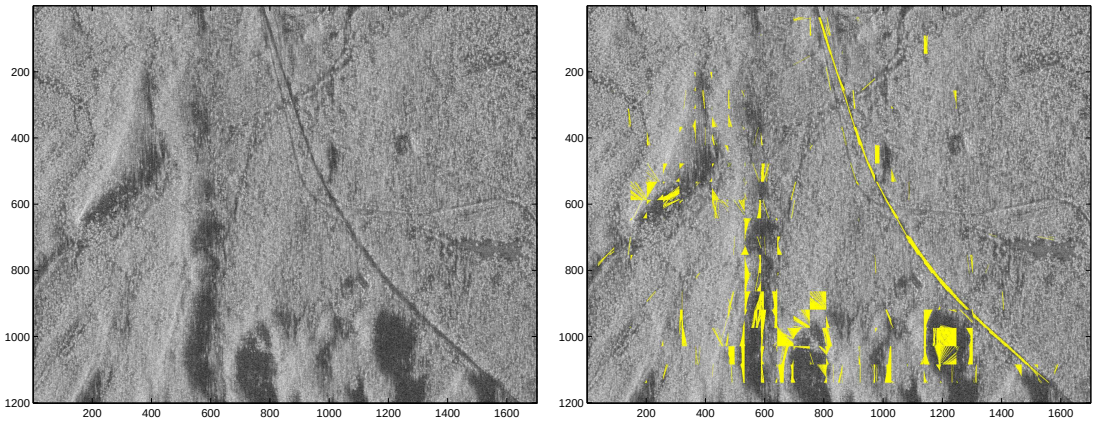


Figure 1.2: *The best results for the hypothesis test were obtained assuming no correlation, with* $\lambda_{thresh} = -1.5$

$$\lambda \approx \frac{N}{2}\left[\log\frac{(1-\bar{b}^2)\bar{c_0}}{(1-b^2)c_0} - \frac{(1-2b\bar{b}+b^2)\bar{c_0}}{(1-b^2)c_0} - \frac{(\bar{m}-m)^2}{c_0}\left(\frac{1-b}{1+b}\right) - 1\right] \tag{1}$$

where N is the size of the sample on each line, m is the mean, and the overline corresponds to the value of the statistic on the line (while without the overline refers to the background). The mean and the correlation coefficients for each line are calculated with the aid of a Radon transform in a sliding window arrangement as described in section 1.2.

Surprisingly, this more realistic model for the pixel intensity distribution generates results which are significantly worse than for the simple uncorrelated case. A number of reasons suggest themselves as to why this is the case. Firstly, after examination of lines along a number of faint trails in SAR images, it was found that the correlation between neighbouring pixels was not significantly different from that of the background. For this reason, the correlated hypothesis test would essentially be determining whether the chosen line differed significantly from the mean of the distribution, which is what the uncorrelated case was doing already.

Also, the correlation coefficient of background lines were measured and found to vary enormously, especially in lines containing large intensity gradients (such as lines perpendicular to roads). As a consequence, the background lines with much higher or lower correlations are much more likely to be over the maximum likelihood threshold, and so will increase the FAR, while not significantly affecting the PD.

Due to these factors, it is not expected that a method based on correlated hypothesis testing would produce good edge detection rates in SAR imagery.

1.3.2 Karhunen Loeve Transform

The discrete Karhunen Loeve Transform is often used in signal processing [1] for aligning objects with their axes of symmetry. These axes of symmetry are calculated by de-

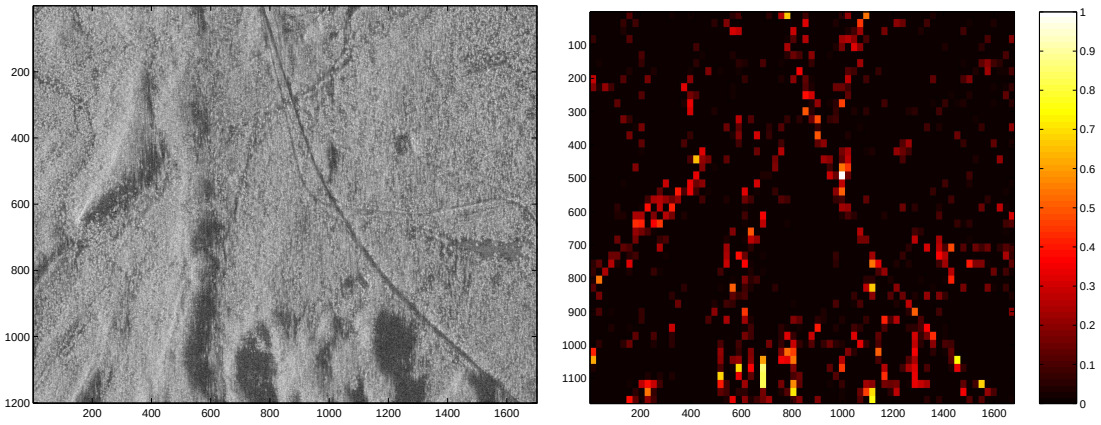


Figure 1.3: *Using the Karhunen-Loeve based method. The colour roughly corresponds to the probability a block contains an edge.*

termining the eigenvectors of the covariance matrix formed by taking the coordinates of each of the points belonging to the object concerned. For the purposes of edge detection, this idea has been modified as follows.

A particular block of an image which contains an edge will tend to be oriented either in, or perpendicular to the direction of the edge. Since blocks corresponding to background will usually also have a preferred orientation (although to a lesser extent), some sort of measure for the degree of 'directedness' of the block was necessary to distinguish the blocks of interest from the background. The measure used was the ratio of the eigenvalues of the covariance matrix obtained from the coordinates of points in the block, which had been weighted by the grey level intensity of each point. Blocks having a higher eigenvalue ratio were deemed to be more directed and hence more likely to correspond to an edge.

This algorithm was run using size 24×24 blocks, and each block was given an intensity according to that block's calculated eigenvalue ratio, so the lighter blocks correspond to those more likely to contain an edge. After applying the method to several images, it was found that blocks containing edges were detected well for main roads, but very poorly for faint trails, as can be seen in Figure 1.3

1.4 RDRT Method

Figure 1.4 shows a graph of the Radon transform $R(\rho, \theta)$ of a subimage, as a function of one of its parameters ρ, for a fixed angle θ. The large dip in the graph corresponds to the presence of a track having a direction θ. Hence the edges of the road correspond to a high Radial Derivative of the Radon Transform (RDRT), which for the case of a continuous 2D image $f(x, y)$ is given by

$$\frac{\partial \mathcal{R}(\rho, \theta)}{\partial \rho} = \frac{\partial}{\partial \rho} \int_{r=-\infty}^{\infty} f(\rho \cos(\theta) + r \sin(\theta), \rho \sin(\theta) - r \cos(\theta)) dr \qquad (2)$$

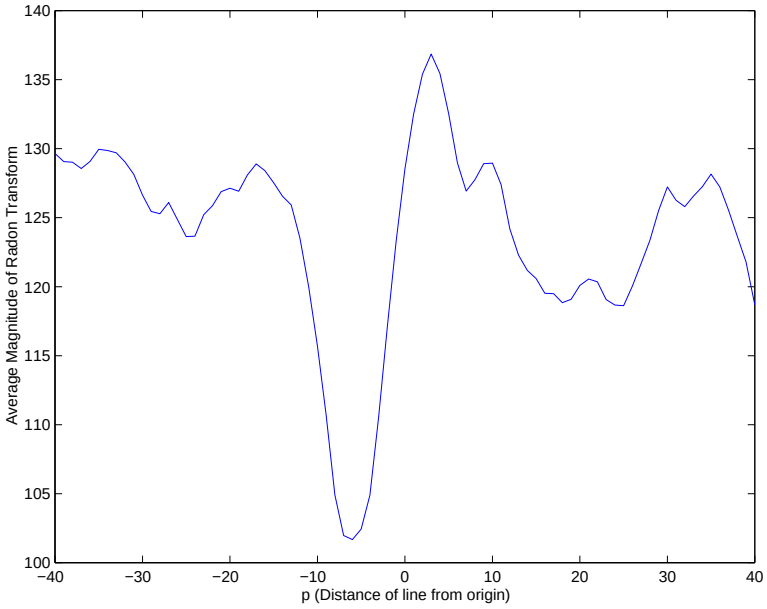


Figure 1.4: *The variation of the Radon transform with ρ for an image containing a faint road for fixed angle θ*

The fact that high RDRTs correspond to road edges can be more easily understood by noting that a large change in the Radon transform due to a small increment in the parameter ρ means there is a large change of average intensity between two close parallel lines. The most likely reason for such a change in intensity is the presence of an edge parallel to these lines. Using this result, the calculation and identification of the appropriate RDRTs may be used to mark line segments that correspond to edges. As described in section 1.2, the calculation of the Radon transforms involved is done in a sliding window arrangement.

A number of different techniques were used to determine which radial derivatives corresponded to edges. In the first instance, a single threshold was used to remove edges from background. This method worked quite well, producing significantly better data than the hypothesis testing scheme in the detection of faint trails. They also detected only the edges of tracks (rather than all lines belonging to the roads) and did not detect extraneous objects such as lone hills which due to the fact that they differed significantly from the background, were detected by the hypothesis testing algorithms.

For high thresholds, the faint tracks were still not detected strongly enough for identification purposes, and for low thresholds there were too many false detections. In order to eliminate these false positives, some other methods were attempted.

The second procedure that was tried found the standard deviation of the RDRTs, and used only those lines at which the derivative differed from the mean by a certain threshold times the standard deviation. These statistically significant lines were assumed to be edges. This method did not seem to be any improvement over the previously used method.

The third procedure tried involved examination of the angular derivative. For straight

line segments, it is expected that the derivative of the Radon transform with respect to the parameter θ be a maximum. Since this is highly dependent on the curvature of the roads, then the more curved roads (which the Radon transform algorithms have great difficulty detecting anyway) become even more difficult to detect. Also, the ability of the algorithm to detect straight roads or ignore false trails was not significantly improved, so no further methods involving calculation of angular derivatives was attempted.

The detection of isolated edges due to thresholding the RDRT seemed to have a high PD for lower thresholds, but a similarly high FAR. In order to improve the results, either an improved PD algorithm or some sort of false alarm mitigation algorithm needed to be used. Since the PD was already quite good, a technique was devised for ignoring false alarms, based on the fact that any roads of interest were likely to be long, and so consist of many line segments joined together. Isolated line segments could be ignored.

1.4.1 Linking Algorithms

There are a number of techniques detailed in the literature concerning the linking of edge elements to form a continuous edge. One such method is stochastic relaxation, the basis of which was first outlined in Rosenfeld, Hummel and Zucker [5]. More recently Tupin et. al. [6] used stochastic relaxation for road detection in SAR images with good success. Their method however uses simulated annealing which generally proves rather slow, so more computationally efficient techniques were considered.

The first of the attempts at linking Radon edge lines together used a shifting property of the Radon transform. Given two Radon blocks, with centers at $\underline{x}_1$ and $\underline{x}_2$ as shown in Figure 1.5, the Radon transform of the second block in the coordinates of the first is given by

$$R_2^{(1)}(\rho, \theta) = R_2^{(2)}(\rho - (\underline{x}_2 - \underline{x}_1)^T \underline{n}(\theta), \theta). \tag{3}$$

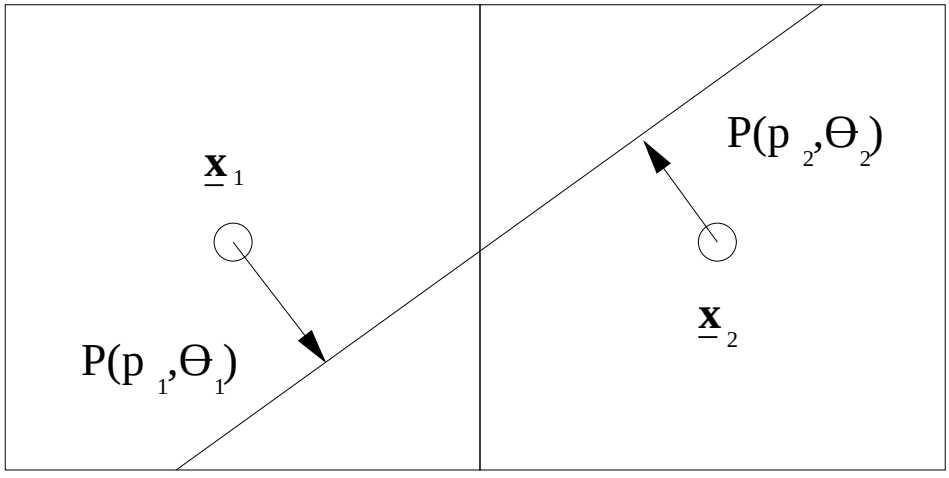


Figure 1.5: *Two neighbouring Radon blocks*

where $\underline{n}(\theta) = (\cos(\theta), \sin(\theta))^T$ is the unit vector in the direction of the angle θ. Using this method, suppose the probability that the Radon line shown in Figure 1.5, which exists in two separate Radon blocks, is a part of an edge. This would be expected to correspond to the product of the probabilities that each of the individual line segments in each of the blocks belonged to edges, that is

$$P(\rho, \theta) = P_1^{(1)}(\rho, \theta) P_2^{(2)}(\rho - (\underline{x}_2 - \underline{x}_2)^T \underline{n}(\theta), \theta). \tag{4}$$

Estimates for the probability functions $P_1^{(1)}(\rho, \theta)$ and $P_2^{(2)}(\rho, \theta)$ may be obtained by considering the associated RDRT for that Radon line. A larger RDRT will correspond to a larger probability that the line is an edge (and a smaller RDRT indicates a smaller probability). After calculation of the probabilities that each pair of line segments (corresponding to the same line) belong to an edge of interest, those pairs that are above a certain threshold probability may be used to detect the presence of edges.

This method of detection turned out to be less successful than before. A variation on this technique was then considered which took possible variations in position and orientation of lines segments in neighbouring Radon blocks into account. This procedure added an extra term to 4, which scaled the probability that both lines belonged to an edge by a factor related to the conditional probability that two lines known to be edges in neighbouring Radon blocks, belonged to the same edge. Hence the probability two lines given in Radon coordinates by (ρ_1, θ_1) and (ρ_2, θ_2) formed an edge was given by

$$P(\rho_1, \theta_1, \rho_2, \theta_2) = P_1^{(1)}(\rho_1, \theta_1) P_2^{(2)}(\rho_2, \theta_2) F(\rho_1, \theta_1, \rho_2, \theta_2) \tag{5}$$

where the function F was a function of the distance between the two lines and the change in the orientation from one to the other. The function was chosen to satisfy the criteria that the smaller the difference between the two lines, the more likely they were to belong to the same edge.

It was hoped that this method for linking lines in consecutive Radon blocks would prove much more effective than the previous attempt, but trial of this method did not significantly increase the PD of the method. Another method for linking was produced later however, which yielded both a much higher PD and a lower FAR.

1.4.2 Hysteresis Thresholding

While the thresholded RDRT detection method for high thresholds seemed to detect parts of roads very well with a low FAR, on the fainter roads the detection was often sparse. Hysteresis thresholding capitalises on this low FAR rate by introducing two thresholds. The upper threshold was used to obtain line segments in the image which are very likely to belong to paths, and these paths are continued until the RDRT decreases below the lower threshold.

In more detail, each line segment having an RDRT higher than the upper threshold is assigned a path number. For each of these lines, the intercepts with the sides of the Radon blocks are determined, and so the neighbouring block which contains the extension of the

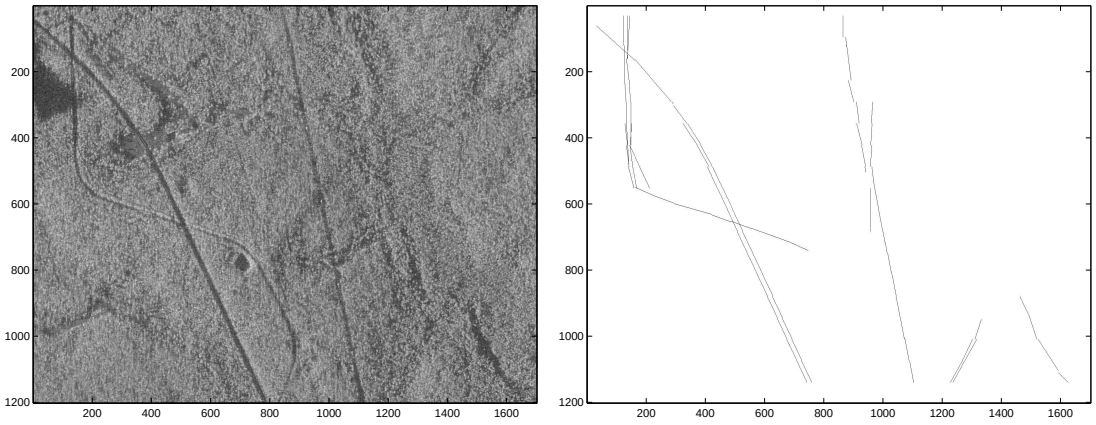


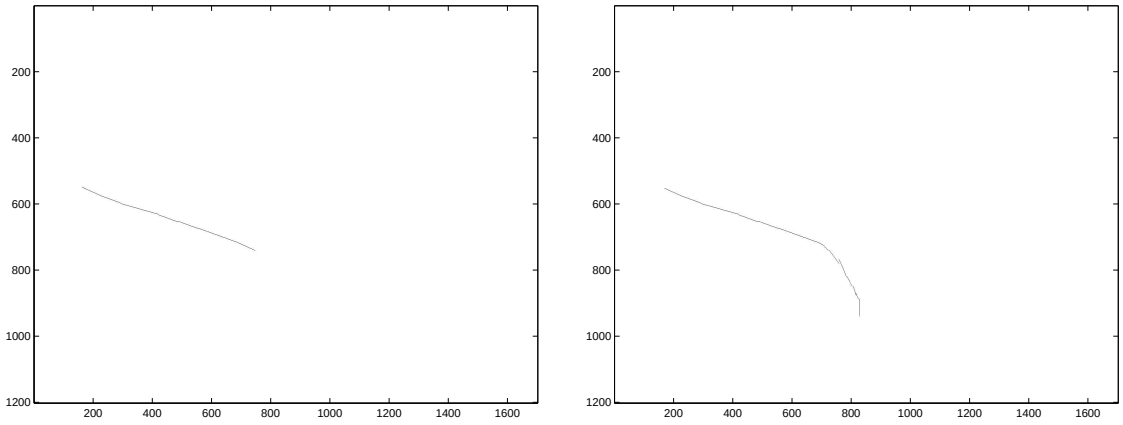
Figure 1.6: *Using the hysteresis threshold RDRT method for* $t_u = 5.5$ *and* $t_l = 3.3$. *Part of the trail is not well detected.*

line segment is found. The Radon coordinates of the line in the new block are determined from the shifting theorem. Since the path being searched for is not necessarily straight, the neighbourhood of the line in Radon space (which corresponds to lines having similar position and orientation) are examined for lines having high RDRTs. If the RDRT of this neighbouring line is above a specified lower threshold, then the line having this property is added to the path, and the process continued. Otherwise the path is assumed to stop.

Due to the low FAR of the high threshold detection, the path is only started at lines which are almost certain to be a part of a road or track. The lower threshold which determines when the path ends can then allow the road to be detected even over regions where the edges are less well defined. Since the lower threshold lines of the path are only added to the end of lines that are almost certainly roads, this reduces the FAR which is normally associated with a low threshold without significantly compromising the PD. Removing paths which are below a certain length may also help reduce FAR images, since shorter line segments are less likely to correspond to tracks of interest.

1.4.3 Multiscale Algorithm

The above method had good success with detecting long and straight tracks and trails. It was however much less successful at the detection of curved segments of road, as shown in Figure 1.6 and could not always detect short straight parts of faint trails separated by highly curved components. This was because the initial high threshold was not guaranteed to detect the short components of the road due to it's lower PD per length of road scanned (while it it almost guaranteed to detect a portion of an extremely long road), and due to the large scale of the Radon blocks, the curved portions of the road cannot often be followed since at that scale the road is not well approximated by straight line segments. It is therefore desirable to use a much smaller block size for trail detection near corners. The size of the blocks should not be made too small however since otherwise the path would be extremely susceptible to false alarms and would be extremely likely to leave the road completely.



(a) Without Multiscale (b) With Multiscale

Figure 1.7: *One particular path from Figure 1.6, with and without using multiscales*

Using this argument, code was developed which, when the end of the path was reached at the larger block size, switched to a block size about half the size of the original. This appeared to have some success in allowing curved sections of road to be located, as shown in Figure 1.7. The resulting detector had a very good PD and FAR rate.

1.4.4 Calculation of False Alarm Probabilities

The likelihood that any road produced by the above method can be related to the average RDRT over the path. The higher the RDRT, the sharper the edge and the more likely that it belongs to a feature of interest. The exact relation of the average edge intensity to the probability of the path actually belonging to a trail may be obtained by

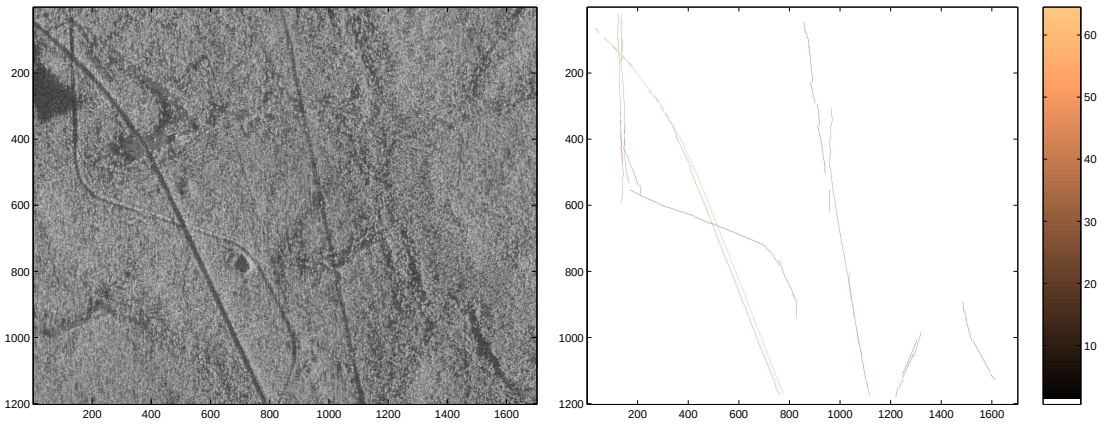


Figure 1.8: *Multiscale hysteresis RDRT detection. The brighter colours correspond to sharper edges.*

experiment. Figure 1.8 shows edge detection, including average edge intensity. The main roads are much brighter than the faint roads, which are in turn much brighter than the false alarms.

1.5 Pseudo-code

Presented below is one possible pseudo-code realisation of the RDRT algorithm described earlier. Some alterations which could be made to this pseudo-code are outlined in the next section on computational considerations.

1 **Initialization**

Initialise variables, set algorithm parameters (thresholds, line lengths and subimage and window sizes) and calculate the number of windows required to cover the screen (see Figure 1.1).

2 **Calculation of the RDRTs**

For each of the windows in the image, calculate the RDRT blocks of the associated screen limited to the window in the transform domain.

3 **Find high threshold RDRT lines:** Set rl, the set of Radon lines of interest, to the empty set ϕ. Then for each of the RDRT blocks calculated in Step 2, perform the following

A) **Thresholding:** Find all those Radon lines which are above the upper threshold set in Step 1. For each of these Radon lines do the following

a) Check if the Radon line *really* lies within the window (since Step 2 only very roughly confined the RDRT within the window). If it is, then add the line to the set rl, otherwise ignore it.

B) **Thinning:** For each Radon line now in rl, check whether it is within a certain neighbourhood (specified in Step 1) in Radon space of another line in rl. If it is, then these two lines probably correspond to the same edge, so the one with the highest RDRT is kept while the other is removed from rl.

4 **Finding paths:** Initialise *path*, the set of all Radon lines belonging to a connected edge, to the empty set ϕ and set the path number equal to 0. For each line belonging to rl, initialise a path as follows

A) Add one to the path number, and add the line from rl to the path. This line is used to start the path, so for each direction follow the path until the RDRT decreases below the lower threshold. This is performed by setting the variable *stop* to zero and the 'previous line' variable to the line from rl, then repeating until *stop* is 1 the following

a) Find the side through which 'previous line' leaves, and obtain the neighbour block which shares this edge. If this block is off the edge of the image, then the path stops and *stop* is set to 1 and the loop ends. Otherwise the path continues into this new block.

b) Find the Radon coordinates of 'previous line' in the new block by using the Radon shift theorem (equation 3).

c) Focus on the RDRT of this block (obtained from Step 2) in a small neighbourhood (specified in Step 1) of the previous line in the new Radon block. If the maximum RDRT is less than the lower threshold (also specified in Step 1), then this is the end of the path and *stop* is set to 1 and the loop terminates. Otherwise it continues.

d) Check whether the line corresponding to the maximum RDRT in the neighbourhood *really* lies inside the window.

e) If the Radon line lies inside the window, then check if it is already part of the path, in which case the path has doubled back on itself and the loop should stop with $stop = 1$. If the path hasn't doubled back on itself, then change the 'previous line' variable to the new Radon line and add it to *path* (along with the path number) and leave $stop = 0$.

f) If the Radon line doesn't lie inside the window, then the previous line has continued into a diagonal neighbour from it's original block instead of an edge sharing neighbour. Change the 'previous line' variable to the new Radon coordinates so that the path will continue the next time through the loop.

B) Now that the entire path associated with the line from *rl* has been determined, then find the length of the path by finding the number of elements of *path* having the current path number.

C) If the length of the path is below the length specified in Step 1, then remove the entire path from *path*.

D) Remove all those lines from *rl* which have not yet been considered in Step 4 and are a part of the current path. This helps prevent duplication of a path.

5 **Finer Grid Scale:** Each of the paths calculated in Step 4 may have terminated due to a sharp bend which the coarse Radon line size may have been unable to handle. As a consequence, for each Radon line which is two line segments from each end of each path (which is set to the 'previous line' variable) do the following.

A) Set $stop = 0$, then repeat the following until *stop* is set to 1.

a) Calculate where the 'previous line' intersects with the boundary of its Radon block. Then define a new fine size Radon block which shares the edge of the first Radon block where 'previous line' intersected, and whose center lies level with the intersection point, as shown in Figure 1.9. This ensures that the continuation of the path would definitely lie within this new Radon block (unlike in Step 4A, where the path may in fact have continued into a diagonal neighbour), thus preventing difficulties with corners.

b) Find the Radon coordinates of 'previous line' in the new Radon block by using the Radon shift theorem.

c) Calculate the RDRT for the new Radon block, and search the neighbourhood (whose size was specified in Step 1) of the new Radon coordinates of 'previous line'. If the maximum RDRT in this neighbourhood is less than

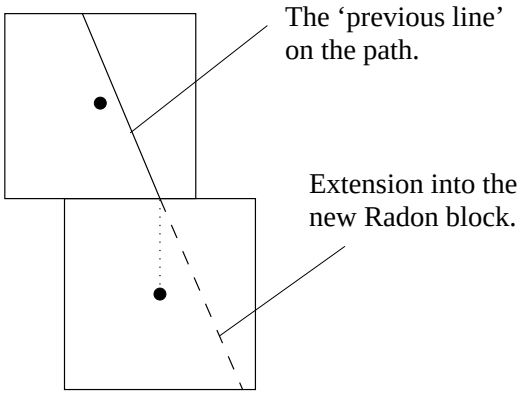


Figure 1.9: *Defining the new Radon block for step 5A of the pseudo-code*

the lower threshold for the fine scale (also specified in Step 1), then the loop terminates. Otherwise it continues.

d) Set the new 'previous line' to be the line having maximum RDRT in the neighbourhood of the old 'previous line'.

6 **Output**

For each of the paths generated, average over the RDRTs for each line in the path (scaled so that the smaller lines contribute proportionally less to the average). Then plot the entire path with a colour which corresponds to that average edge intensity.

1.6 Computational Considerations

The pseudo-code in the previous section calculates the Radon transforms of the individual blocks only once during the determination of the lines satisfying the upper threshold of the RDRT. During the linking component of the procedure (at least for the larger block size), the RDRTs that are needed are just read from memory instead of being recomputed. While this is an enormous time saver, it requires that the Radon transform of the entire image be saved in memory, which is quite a significant expense. This could be improved by either applying the algorithm to smaller blocks of the image, or since the majority of the image will not contain roads at all, by recomputing the Radon transform of each block during the linking stage (this is already done for the smaller block size). Another advantage of this would be a slightly improved detection performance, since the joining block would not necessarily need to already exist in memory. This would allow difficulties concerning the road passing through corners of blocks to be circumvented by utilising the pseudo-code version of the finer grid scale (Step 5 of the pseudo-code) in place of the current rough grid algorithm.

Another issue of relevance is the algorithm used to calculate the Radon transform. The MATLAB implementation uses the built in Radon transform procedure, which is extremely inefficient. Julian Magarey of CSSIP's fast Radon transform method [4] is able to generate an $N \times N$ Radon transform with a computational complexity of $N^2 \log(N)$, but this method

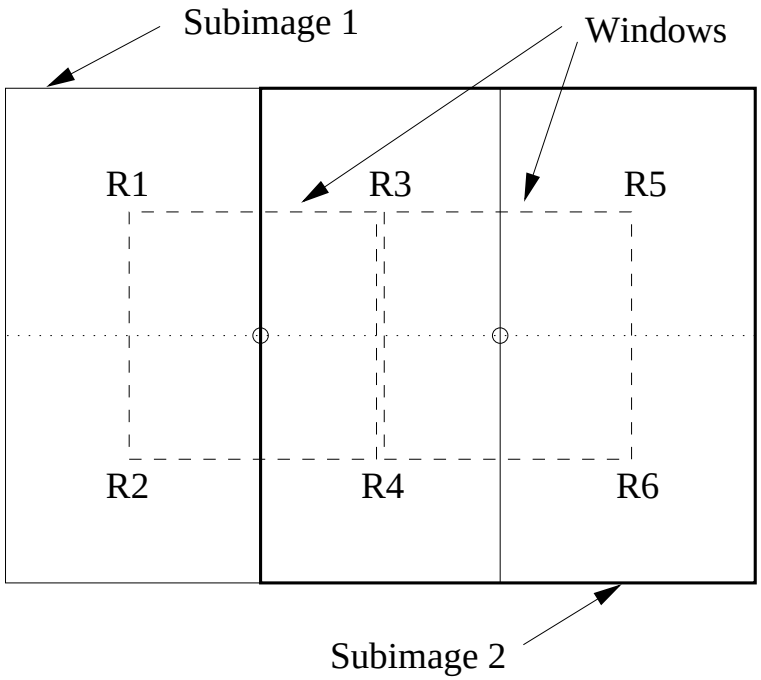


Figure 1.10: *Subimage and window geometry when the window size is half that of the subimage*

requires an image size corresponding to a power of 2. Since the Radon transforms are being calculated over small subimages, it is not expected that the improvement in speed obtained by using this algorithm will be hugely significant, but the possibility should be kept in mind.

The optimal subimage and window sizes have been found to be approximately 128 and 55 respectively (although a window size of 64 has been used in all of the examples presented in this report). Since the Radon transform is being calculates over the subimage, and then limited to the window, then Magarey's algorithm may be applied for the optimal case. Some extra properties however may be used to improve efficiency if the window size is also a power of 2. Magarey's algorithm uses a quadtree structure to generate the Radon transform of a larger block in terms of two smaller blocks. From figure 1.10, it can be seen that when the window size is a half the subimage size, the Radon transform of the first subimage is calculated from R1,R2,R3 and R4. Since R3 and R4 have already been calculated, they may be used to calculate the Radon transform of the second subimage, with only the added calculation of R5 and R6. This effectively halves the calculation time of the Radon Transforms. This speed improvement comes at a cost of a marginal decrease in the PD, since the optimum window size does not appear to be a power of 2.

1.7 Testing

The algorithm developed previously operates based on a number of parameters, namely the high and low thresholds for the coarse scale RDRT, the low threshold for the fine scale RDRT and the low threshold for the average edge intensity. Four 1000×1000 images containing what seems to be (since groundtruth is not known) grassland, woodland, hills, a main road and some faint trails were used as a training set on which to manually tune the parameters. Good detection occurred over a range of these parameters, and six different sets of parameters were produced to be tested.

The RDRT thresholds were split into two types. One set of parameters was 'sensitive', in that it was more likely to detect non-linear features other than roads, which may be of interest. This corresponded to coarse scale RDRT thresholds of 5.5 and 3.3, with a fine scale RDRT lower threshold of 4.5. The other set of thresholds was 'insensitive', having a greater confidence at detecting roads but being less likely to detect other curvilinear features. This corresponded to coarse scale RDRT thresholds of 7.1 and 3.7, with a fine scale RDRT lower threshold of 4.3.

For each of the above RDRT thresholds, a set of three average edge intensity ranges were defined: a low FAR, a medium FAR and a high FAR range. The 'insensitive' parameters yielded best results with edge intensity lower limits of $3.7, 5.7$ and 6.2 respectively, while the 'sensitive' parameters worked best with lower limits of $3.3, 4.8$ and 5.3.

The RDRT thresholds had been obtained by tuning to maximise the probability of detection in the training images, while the edge intensity ranges were obtained by minimising the FAR, while maintaining the probability of detection.

Once the parameters for the algorithm had been obtained, the algorithm was tested for each set of parameters. The testing procedure was performed using twenty one 1000×1000 images (extracted from two separate larger images). Each image was first examined by eye, and the curvilinear features in each were drawn on a sheet of paper. The features were then grouped into four groups. The group of 'definite' features consisted of main roads and faint trails which the detector is expected to notice. The second group consisted of 'ultra fine' features, rivers and creeks which are noticeable by eye, but are not necessarily of interest (and for which it is not claimed that the algorithm would detect. It would be considered a bonus for the algorithm to detect these features). The third group consisted of curvilinear features (denoted 'linear' in the table) which are not easily identifiable by eye.

After the grouping, the algorithm was run over the images for each of the six sets of parameters. The output from the program for each range and each set of threshold parameters was compared against the output that had been drawn on the paper earlier. From this, measures of the detection rate and the false alarm rate for each of the combination of threshold parameters and edge intensity ranges were obtained. These results are presented in Tables 1.1 and 1.2. An example of the output of the procedure using sensitive parameters has already been shown in Figure 1.8.

The FAR for the 'sensitive' parameters as given in Table 1.2 seems inordinately large, however the majority of these false alarms were obtained in the final three images, which contained a river. Since the detector operating using 'sensitive' parameters had previously

Table 1.1: *Measurements of algorithm detections using 'insensitive' parameters*

	Feature Type			
Detector	Definite	Ultra-fine	Linear	False Alarms
Eye	9	43	22	0
High FAR	9	20.5	2.5	3.5
Med. FAR	9	18	2	2.5
Low FAR	7.5	13	1	1.5

Table 1.2: *Measurements of algorithm detections using 'sensitive' parameters*

	Feature Type			
Detector	Definite	Ultra-fine	Linear	False Alarms
Eye	9	43	22	0
High FAR	9	30.5	5	32.5
Med. FAR	9	30.5	3	24
Low FAR	9	26.5	3	9.5

detected a number of features that had not been detectable by eye before being brought to attention by the computer, it is possible that the detector is picking up the previous courses of the river. It is also probable that better results could be obtained if a wider variety of terrain had been included in the training set which for this test did not include river land. In any case, the FAR was much lower in the eighteen images which did not contain river, giving $16.5, 11$ and 3 as the number of false alarms for the high, medium and low FAR edge-intensity ranges respectively. The three images containing river contributed to more than half of the false alarms measured.

It should be noted that the grouping of the features used in this experiment were not produced by an image analyst, who would know which features were considered of interest. Also, the training set used for tuning the parameters was reasonably low, and so the statistics presented here should only be considered a rough guide until further testing can be performed at DSTO.

References

1. Gonzalez R.C. and Woods R.E., *Digital Image Processing* Addison-Wesley, Reading, 1992.

2. Newsam G.N., Line detection as an exercise in hypothesis testing, Draft paper.

3. Steger C., An unbiased detector of curvilinear structures. Technical report, Forschungsgruppe Bildverstehen(FG BV), Informatic IX Technische Universität München, 1996.

4. Magarey J.F.A. and Newsam G.N., A fast hierarchical algorithm for computing the Radon transform of a discrete image, In preparation.

5. Rosenfeld A., Hummel R.A. and Zucker S.W., Scene labelling by relaxation operations, IEEE Transactions on Systems, Man and Cybernetics, Vol6. No6. June 1976, pp.420-433.

6. Tupin P., Maître H.,Mangin J., Nicolas J. and Pechersky E., Detection of linear features in SAR images: Application to road network extraction, IEEE Transactions on Geoscience and Remote Sensing, Vol36, No2. March 1998, pp.434-453.

Chapter 2

Feature Extraction I

2.1 Introduction

The main focus of the JP129 project is to provide an automated system to help analysts to detect targets of military significance. The sheer amount of data in a single SAR image completely overwhelms the small amount of target data, and high variation in background clutter along with small target size, low resolution and limited signal to noise ratios make finding targets a time-consuming task for human analysts. The system currently in use is a simple detector where individual points in the image which produces a radar return statistically stronger than its background are passed to the analyst for assessment. The false alarm rate (FAR) produced by this simple detector however is extremely high, and the analyst is still required to classify an enormous number of candidate targets.

The methods outlined in this report provide techniques for reducing the number of false targets that the analyst is required to classify, while ensuring that as few actual targets as possible are discarded.

2.2 The Data Sets

The testing of the techniques developed in this report were performed over a variety of data sets. The first of these data sets consisted of a series of three runs of single look complex data with non-square pixels. Run 1 contained complex matrices of size 22×22, 53 of which corresponded to background and the remaining 159 containing high contrast targets. The second and third runs were of a similar format, but contained only known targets (300 in the second run and 327 in the third).

The second data set also contained complex data, corresponding to 256×256 images of grassland, woodland, riverland and floodland. Data sets one and two were the only complex data sets available, and they are only used to give estimates for the improvement in detection rate that could be obtained if complex data were available at the groundstation, instead of only the magnitude data. For more accurate values of PD and FAR for each detector, larger data sets containing only magnitude data were considered.

The third data set consisted of magnitude only SAR data, and was obtained by running the existing prescreening algorithm over a 5 separate large images, each containing inserted targets as explained in Redding [3]. The data was labeled as a 'hit' or detection if the prescreening algorithm detected an inserted target, and a false alarm otherwise (although since groundtruth is not known, some of these may actually correspond to true targets, but this is expected to correspond to a very small percentage of the total). Each 64×64 area flagged by the prescreening process as containing a target was then sorted into groups depending on a measure of the contrast and clutter of that region. In total, there were 22084 targets and 53961 background regions in this data set. The majority of the tests performed used this data set.

2.3 Individual Target/Background Features

The usual method for target detection is to determine a number of key target 'features' and to use a method such as a Support Vector Machine (SVM) to determine the best separation of targets from background in this feature space.

Generally, features can be divided into two main categories. The first group contains properties of the target itself. Due to the low resolution of images, it is difficult to see by eye, much more information in the target other than maximum intensity, size and possibly a target orientation. The second group of features concerns the relationship between a target and its immediate background. It is only the differences between target and background which allow a target to be seen at all by eye. This group of features is expected to provide the most useful information for the mitigation of false alarms from the prescreening algorithm.

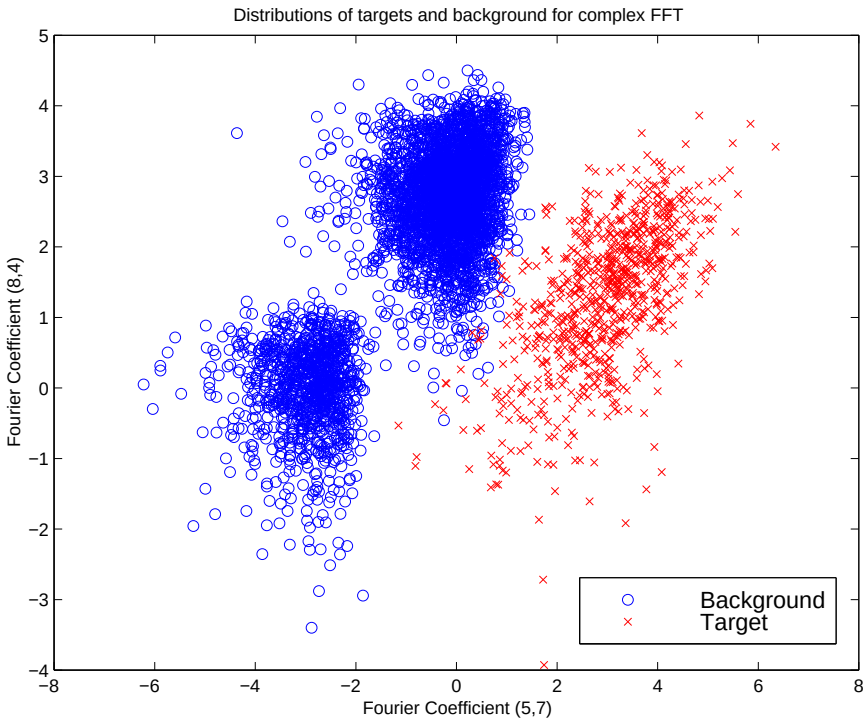
The following subsections define a number of possible features from each of the above two categories. Each of the feature's usefulness in discrimination is discussed in the section concerning combinations of features.

2.3.1 Fourier Coefficients

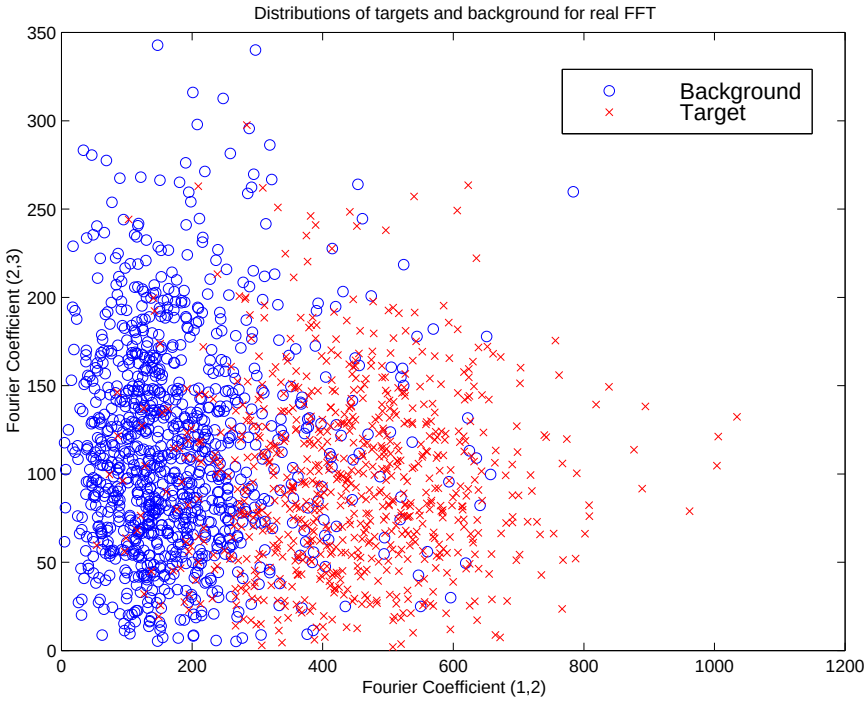
The 2D Fourier transform of an image contains a significant amount of useful translation independent information. The presence of a target in an image should introduce a peak at some frequency dependent on the target, and so certain coefficients should be especially well suited for discrimination between targets and background. The use of FFT coefficients for both real and complex image data is now considered.

2.3.1.1 Complex Data

The return from a radar has both an in-phase and a quadrature component, which may be written as a single complex number. As far as human vision is concerned, it is only the magnitude of this data which is of importance. Consequently, to halve the bandwidth from the plane to the ground station, the phase is discarded and only the magnitude is transmitted. Although the phase component is not useful visually, it still does contain a good deal of information.



(a) Complex FFT Coefficients



(b) Magnitude FFT Coefficients

Figure 2.1: *Distributions of targets and backgrounds for the FFT coefficients of the real and complex images*

It was found after a computer search, that using FFT features on 8×8 complex images from data sets 1 and 2 gave best separation using the $(5, 7)$ and $(8, 4)$ coefficients. A similar experiment on the associated real data gave the best discrimination for the $(1, 2)$ and $(2, 3)$ coefficients. Figure 2.1 shows the distribution of the FFT coefficients of target and backgrounds from data sets 1 and 2. Extremely good separation is achieved for the complex data, and training MATLAB's built in discriminant on all of the test data gave a total misclassification rate of only 0.55 percent. The degree of separation for the real data however is much less, indicating that quite a significant amount of information has been lost by ignoring the phase component of the radar return.

Figure 2.2 gives a more quantitative picture of the difference in detection rates between the real and the complex FFT coefficients by comparing the two ROC curves. The complex FFT feature is so much better that on this scale, it is hardly distinguishable from an ideal detector.

The only complex data available for measurement of the performance of complex FFT features is found in data sets 1 and 2. The targets in this collection generally have a higher Signal to Noise Ration (SNR) than would be expected normally in images. To give an indication of the performance of the complex features in less ideal conditions, the center 3×3 squares of images containing targets were removed and transplanted onto background grassland. The intensity of the targets were then scaled according to the required SNR. In this case, the SNR was defined as $10 \log_{10}(S^2/\bar{E}^2)$ where $\bar{E}^2$ is the average background power and S^2 is the maximum power returned from the target. A plot of the PD against the

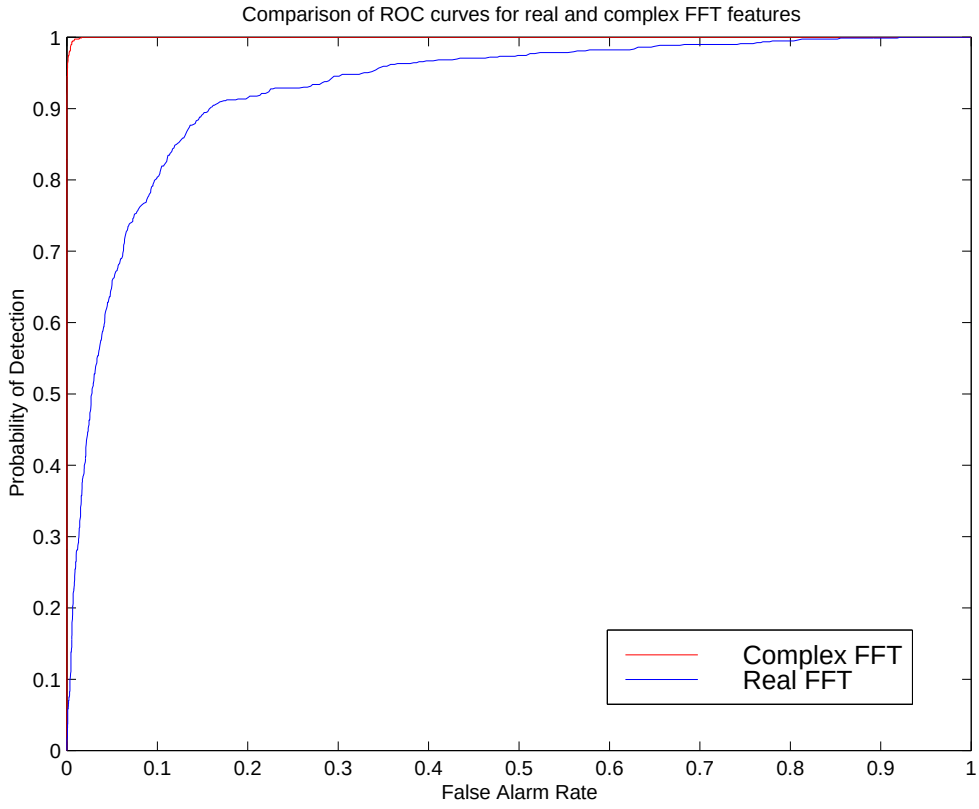
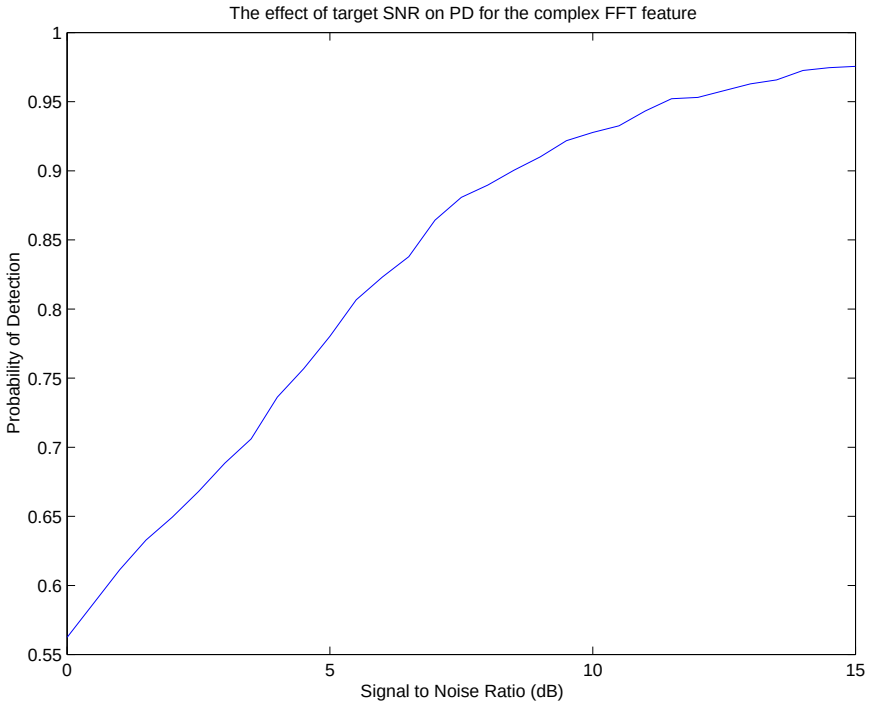
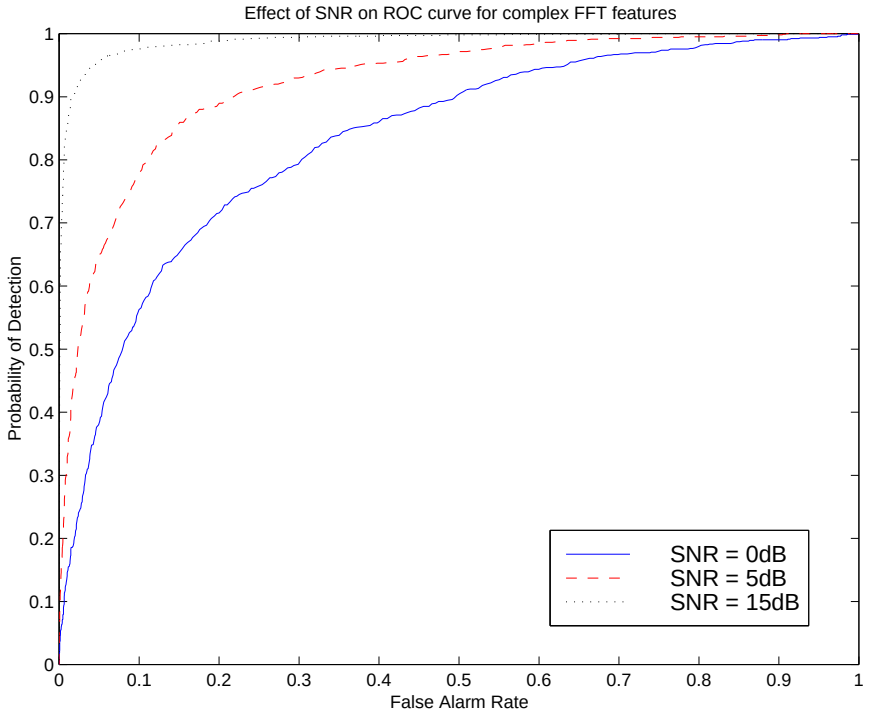


Figure 2.2: *Comparison of ROC curves of FFT features for real and complex data*



(a) PD for a fixed FAR



(b) ROC curves

Figure 2.3: *The effect of SNR of inserted targets on detection characteristics of complex FFT features*

SNR in decibels, for both the complex FFT features, is shown in Figure 2.3. A similar plot would have been generated for the associated real FFT features, but the rough insertion procedure used here seems to have added artificiality to such an extent that even for high SNR data, the detection rate was not as good as for the real non-inserted target data.

Also shown in Figure 2.3 is a family of ROC curves for detection of inserted targets at different SNRs. Although the complex FFT features show remarkable detection characteristics even at low SNR, the insertion procedure does not take into account possible loss of phase information in weak targets.

Due to the lack of available complex data, and the desirability of a set of features based on only the magnitude data, no further testing on complex data is made in the remainder of the report.

2.3.1.2 Magnitude Data

The results shown in Figure 2.1 for the FFT of magnitude data were made, for purposes of comparison, on a small and high SNR data set. To obtain a better estimate for the performance of this method in real situations, the feature was used to classify data set 3 over all clutter/contrast regimes. Figure 2.4 shows the ROC curve for the new data set.

2.3.2 Target Properties

Two features that a target would be expected to possess are a high maximum intensity and for targets larger than a few pixels, some measure of directedness. The measure of directedness used was based on the Karhunen-Loeve Transform (KLT) as described in

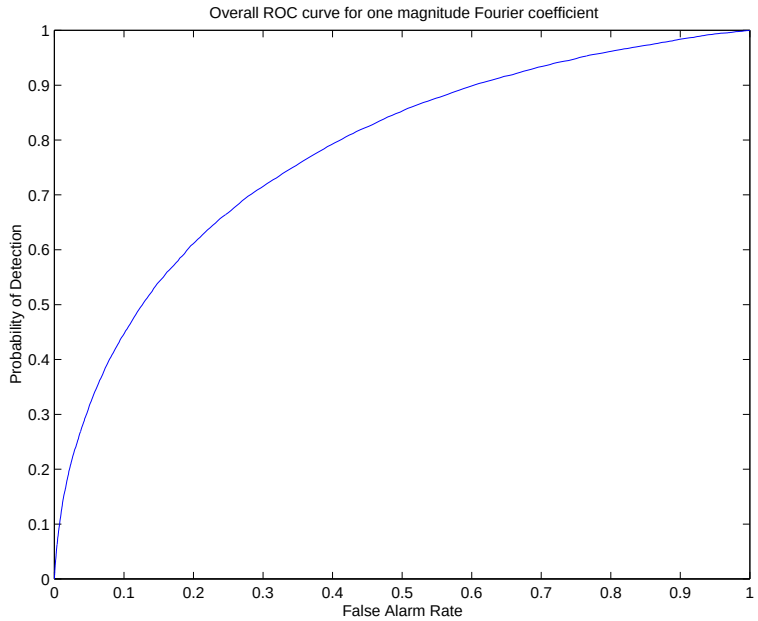


Figure 2.4: *The overall ROC curve for the magnitude FFT feature*

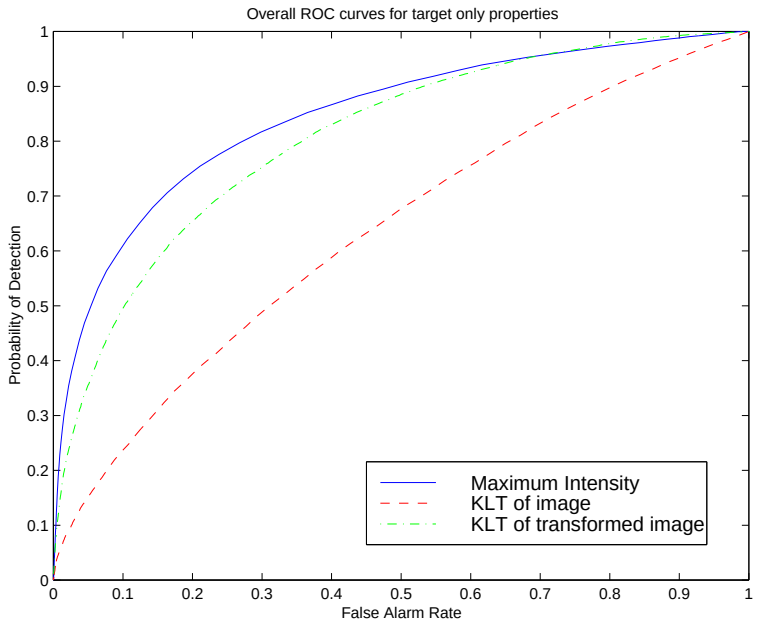


Figure 2.5: *The overall ROC curves for maximum intensity and KLT methods*

Gonzalez and Woods [1]. The KLT finds the principle directions of an image, which are a set of basis vectors which decorrelate the image. These principle directions are evaluated by determining the eigenvectors of the covariance matrix formed by taking the coordinates of each of the points belonging to the image, scaled by its intensity at each point. The strength of 'directedness' of the image would be expected to be related to the ratio of the eigenvalues.

This KLT based method as currently stated, works over the entire image instead of just the target. To increase the effect of target direction, the image may then be transformed by scaling (dividing by 28 seems to have worked best) and then taking the exponential. This has the effect of enhancing high magnitude pixels which are assumed to correspond to the target. Figure 2.5 shows ROC curves for the maximum intensity, and the KLT method on both the original and transformed images.

2.3.3 Correlation Properties

The correlation between a bright spot in an image and its immediate surroundings is expected to be a lot smaller for a target than for a bright patch of background. There are a number of ways of measuring this correlation. For instance:

1 **CorrXY:** Calculating the correlation of rows and columns suspected of passing through a target.

2 **MaxCorr:** The difference in intensity between the brightest spot in the image and the brightest spot of its immediate neighbours. Since both of these points would be

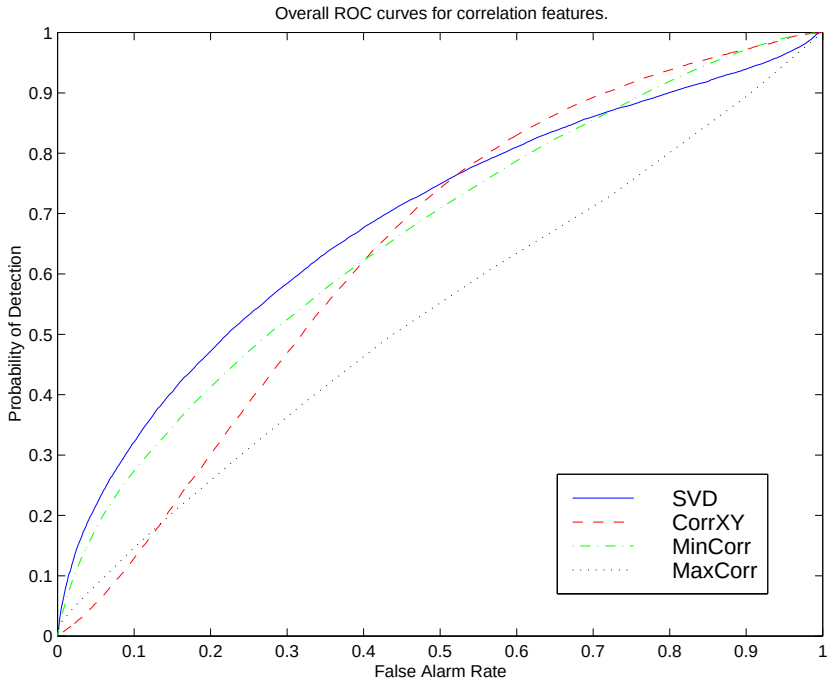


Figure 2.6: *The overall ROC curves for some correlation based features*

expected to correspond to points on the target, this may be considered as a measure of the within target correlation.

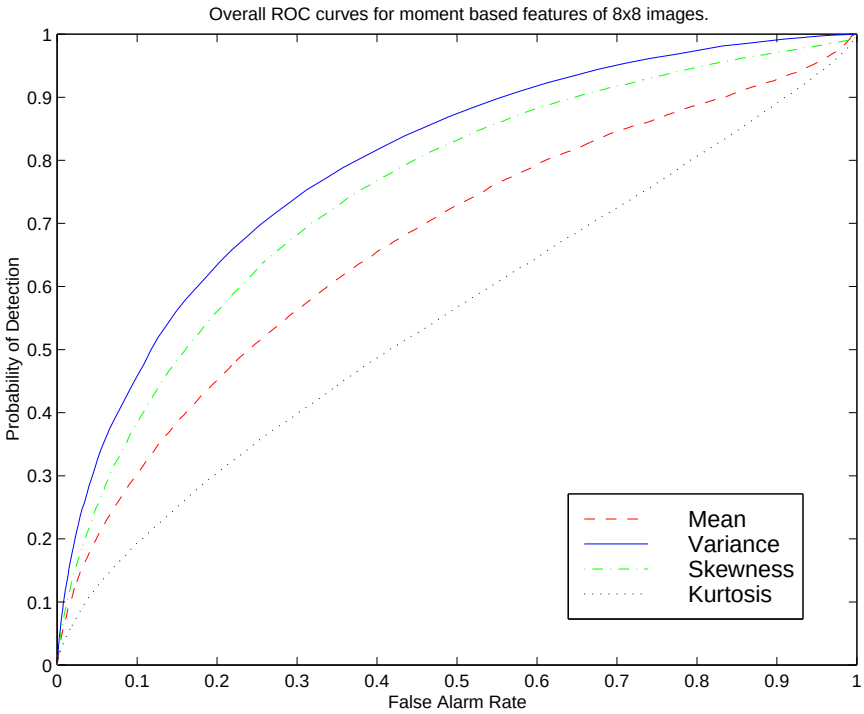
3 **MinCorr:** The difference in intensity between the brightest spot in the image and the dimmest spot of its immediate neighbours. If this is large, it should correspond to a sharp edge which in theory should be more likely to be produced by the presence of a target.

4 **SVD:** The SVD of a matrix $\mathbf{A}$ corresponds to the diagonalisation of its correlation matrix $\mathbf{A}\mathbf{A}^T$. The size of the largest eigenvalue from the SVD of the image could provide useful correlation information for discriminating target from background.

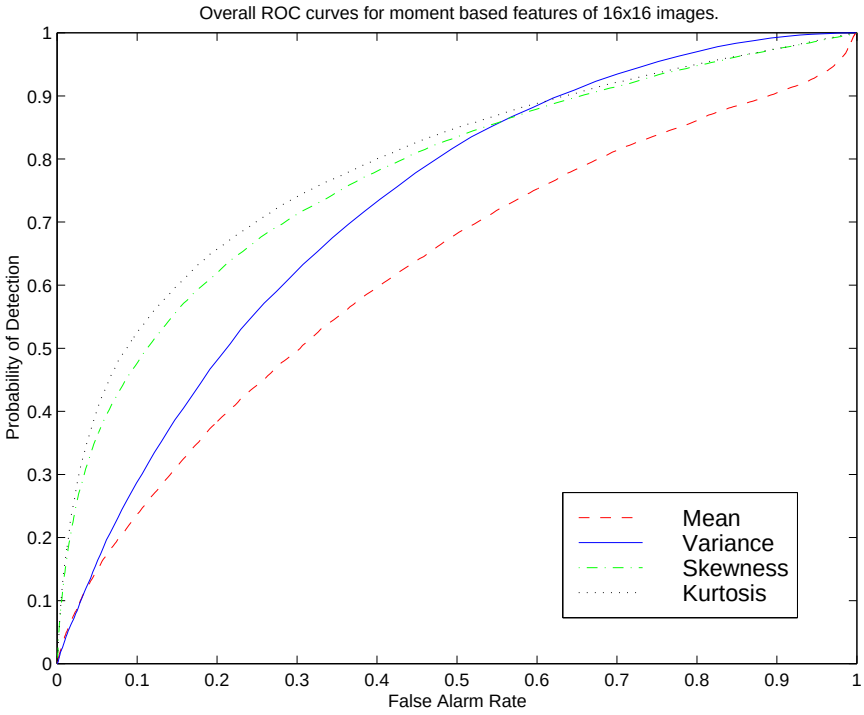
The ROC curves for each of these features are plotted in Figure 2.6.

2.3.4 Changes in Background Distribution

The previous subsection dealt with anomalous local changes in the image distribution. A more global approach to the problem may be taken by considering statistical models for the background, and by assuming that the presence of a target will alter this model. As a result, certain statistical measures of an image could be used to distinguish features from background.



(a) 8×8 images



(b) 16×16 images

Figure 2.7: *The overall ROC curves for some statistical moment features*

2.3.4.1 Moment Features

Theoretically, the probability distribution function of any statistical model can be expressed exactly by specifying all of its moments. In practice however, only a finite number of moments can be calculated, and even then estimation of the higher order moments often have very high errors due to lack of data. More accurate values for these moments could be obtained by increasing the region of interest size surrounding each suspected target, but this also has the effect of decreasing the effect of a target on the moment. In a trade off between discrimination capability and moment accuracy, it was found that the best results occurred for an image size of about 8×8 (although total discrimination ability did not drop off very quickly with increase in image size) and only the first four moments (corresponding to mean, variance, skewness and kurtosis) provided any useful discriminating qualities for this sized image. Figure 2.7 shows ROC curves for each of the four moments, using both 8×8 and 16×16 image sizes.

2.3.4.2 K-distribution Parameters

A fairly common statistical distribution for modeling background clutter is the K-distribution, where the intensity of the radar return at each point is given by the single point probability distribution function

$$p(x) = \frac{2}{x}\left(\frac{L\nu x}{\mu}\right)^{(L+\nu)/2} \frac{1}{\Gamma(L)\Gamma(\nu)} K_{\nu-L}\left(2\sqrt{\frac{L\nu x}{\mu}}\right)$$

where K_n is a modified Bessel function of order n, μ is the mean and ν is the order parameter. Since the presence of a target in the background would have a large effect on the parameters of the K-distribution fitted to the image, these may be used to discriminate target from background.

The major problem with this is that due to the complexity of the distribution function, it is often not feasible to find the best estimate of these parameters from the sample points. Redding [2] discusses however that in most instances, regions of differing K-parameters may be discriminated by using one of three measures

$$\begin{aligned} U &= \log < x > - < \log x > \\ V &= \frac{< x^2 > - < x >^2}{< x >^2} \\ W &= < \log^2 x > - < \log x >^2 \end{aligned}$$

from which the K-parameters can be laboriously estimated (note x is the intensity of the radar return). Since there is just a continuous mapping from these parameters to the estimate for the order parameter, there will be no increase in separation between the two distributions by mapping to the order parameter. As a result, the U, V and W parameters themselves will be used for purposes of discrimination. The ROC curves produced for each of these features are shown in Figure 2.8

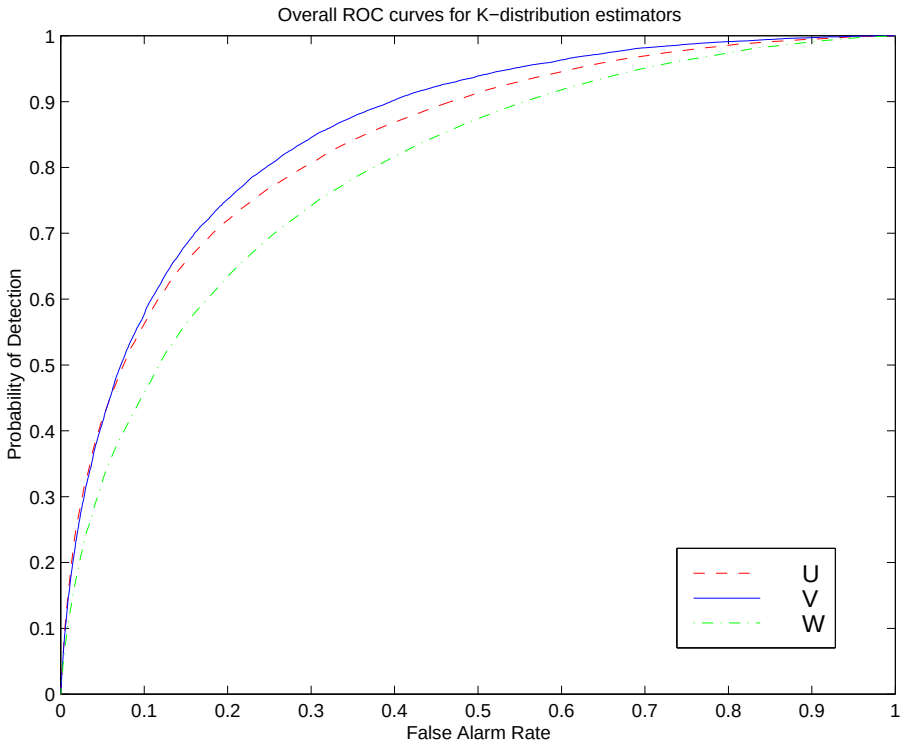


Figure 2.8: *The overall ROC curves for some K-distribution parameter estimators*

2.3.5 Miscellaneous Features

It was found that calculating the Singular Value Decomposition (SVD) of the 2D Fourier transform of the image intensity $\mathbf{A}$ produced useful discriminating features. When $\mathbf{A} = \mathbf{USV}^T$ where $\mathbf{S}$ is a diagonal matrix, then the 2D Fourier transform can be written as $\mathbf{FAF^T} = \mathbf{FUSV^TF^T} = (\mathbf{FU})\mathbf{S}(\mathbf{FV})^\mathbf{T}$. Hence the SVD of the 2D Fourier transform is the equivalent of the 1D Fourier transform of the columns of the matrices produced by the SVD of the original image. The matrix $\mathbf{FU}$ appears to contain the majority of the useful data, but at this stage it is not entirely clear how this physically relates to the target. It is thought however that it may relate to the detection of high energy edge components in the images correlation. The ROC curves for four of the elements of $\mathbf{FU}$ are displayed in Figure 2.9. Although the ROC curve for the $(1, 6)$ coefficient does not appear to be very useful, it can work better in combination with other features as described in Section 2.4.

2.3.6 Summary of Single Feature Results

A comparison of feature performance can be found in Table 2.1. Of the features tested, it appears that the V measure for K-parameter estimation was the best single feature and that the Kurtosis was the worst. The usefulness of each of these features however must be considered in combination with others in order to produce the highest overall detection rate, and this is the topic of the next section.

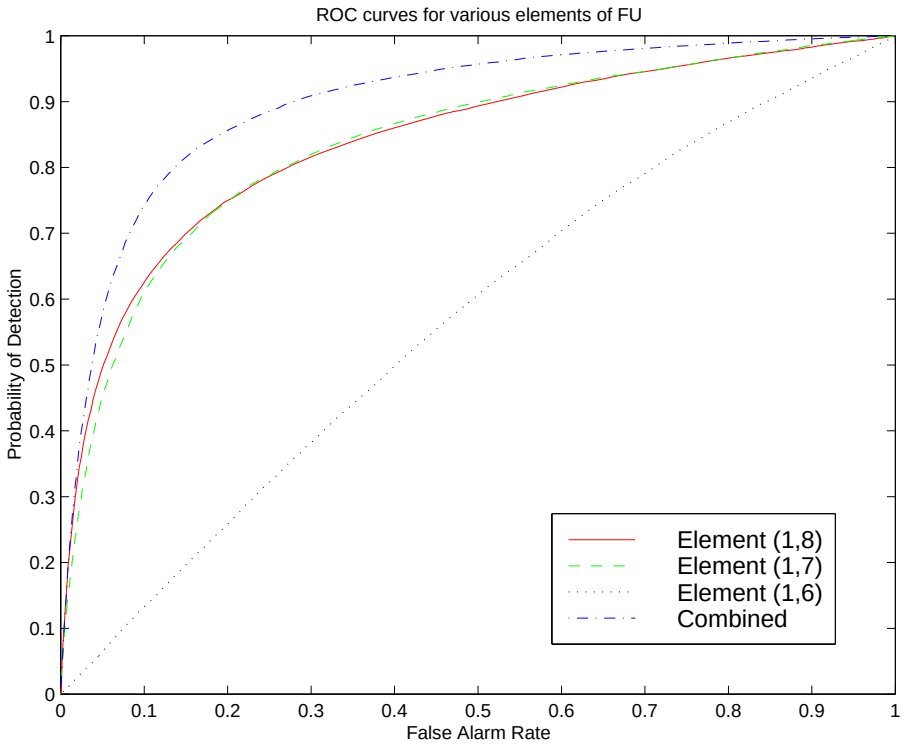


Figure 2.9: *The overall ROC curves for some elements of the SVD of FFT*

Table 2.1: *FAR rates for individual features on* 8×8 *images.*

Feature	FAR at 90 % PD	FAR at 95 % PD
FFT Coefficient	60.6	75.5
Maximum Intensity	51.0	68.2
KLT of exp(image)/28	53.3	68.0
Maximum Correlation	90.0	95.0
Minimum Correlation	77.6	87.7
Row and Column Correlation	70.1	81.6
Diagonal of SVD	80.0	92.5
Mean	83.4	94.3
Variance	57.9	70.9
Skewness	65.2	81.1
Kurtosis	91.5	96.6
U parameter	48.6	63.5
V parameter	39.6	54.4
W parameter	57.9	70.9
Coefficient (1,8) of SVD of FFT	52.5	72.5

2.4 Combined Features

The distributions of target and background distributions in feature space may actually have a better separation than would be expected from observing the projections onto any particular set of feature axes [6]. Features which when taken singly lead to almost no separation, can in some instances produce useful information when combined with other features. The reverse is also true in that features that give useful separation by themselves, may be completely useless when considered in combination with certain other features. In fact, the addition of these extra features can in fact harm the detector performance. The aim of this section is to determine the smallest possible feature set which allows the greatest possible detector performance.

2.4.1 Discrimination and Choice of 'Best' Features

2.4.1.1 Linear Discrimination

When considering the separation of distributions in feature space, the choice of discriminant plays an important role. One simple discriminant is the K Nearest Neighbors (KNN) which classifies each sample point by finding the K points in a training set which are closest to the sample point in feature space. If over half of these points are target, then the sample is classified as a target, otherwise it is classified as background. This method does not allow the generation of ROC curves however, which provide useful information on the separation of the two distributions.

Another simple discriminant is the maximum likelihood test, which models the two distributions as multi-dimensional Gaussians and then thresholds the ratio of the probabilities that a point belongs to the target or the background distribution. In some instances however the decision surface boundaries will not prove robust to changes in the training set. The linear discriminant on the other hand is robust and simple.

There are a number of linear discriminants available. The SVM provides distribution independent linear discrimination, but is extremely slow, having a complexity of order N^3 where N is the number of points to be classified. The Fisher discriminant [4] is an extremely simple and fast linear discriminant, but the separation parameter which it optimises is not connected to the quantities of interest (FAR and PD) in any physically meaningful way. The results presented in this report use a similar linear discriminant which is derived in the Appendix.

2.4.1.2 Choosing the Best Features

Since some of the features listed in 2.3.6 may not improve performance when combined into a system, it is desirable that any redundant features be identified. One usual way of selecting features is to perform a Principal Component Analysis (PCA) which is an eigenvector analysis of the covariance in feature space. These eigenvectors do not always correspond to directions of best separation however. The optimal way to solve the problem would be to examine the performance of all possible combinations of features and

choose the best subset from these. Due to the number of features however, this is not a computationally viable option.

The standard sub-optimal way for finding the best features is to firstly find the best single feature. Then, find the feature that performs best in combination with that first feature, and then the feature that works best with those first two features and so on. The approach taken here is also sub-optimal, but produces a somewhat better result. This method is described as follows

1 **Initialization:** Set a feature set F equal to the empty set. At the end of the algorithm, F will be the set of "best" features for discrimination between target and background.

2 **Loop:** The following steps are repeated.

   A For each possible combination of two features, repeat the following

      a Let G be the union of the two features and the set F. Then calculate the discrimination performance of the set of features in G and store the results.

   B The combination of two features which gave the best discrimination is then considered. If the improvement to the discrimination produced by this combination is not significant, then the loop stops. Otherwise, of these two features, the one which produced the best average discrimination in combination with all of the other features is selected and added to the set F.

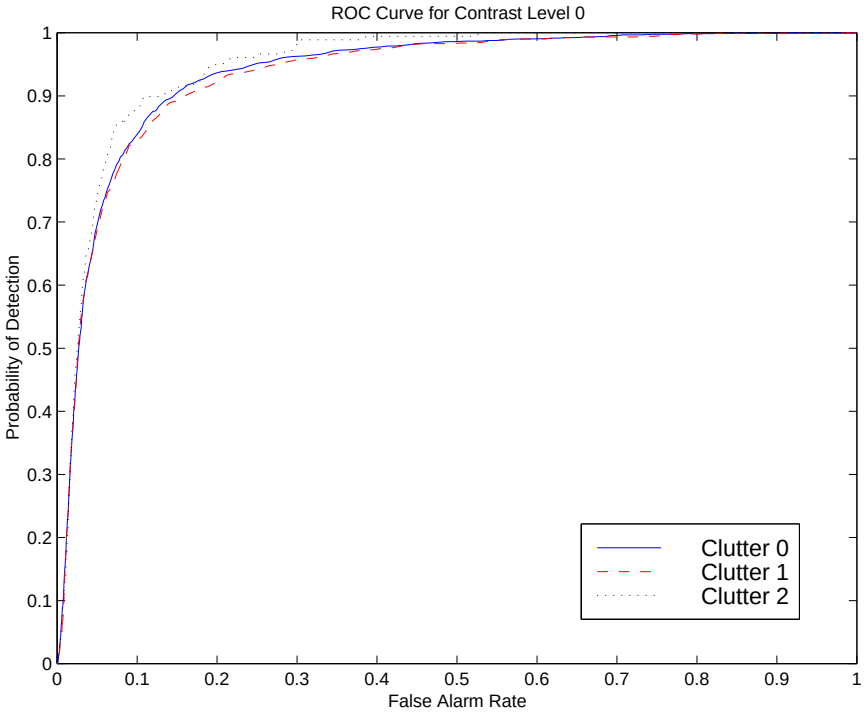
The set of accepted features F produced by this algorithm can then be used to produce a detector having an approximately optimal performance.

2.4.2 Results for Combined Classifier

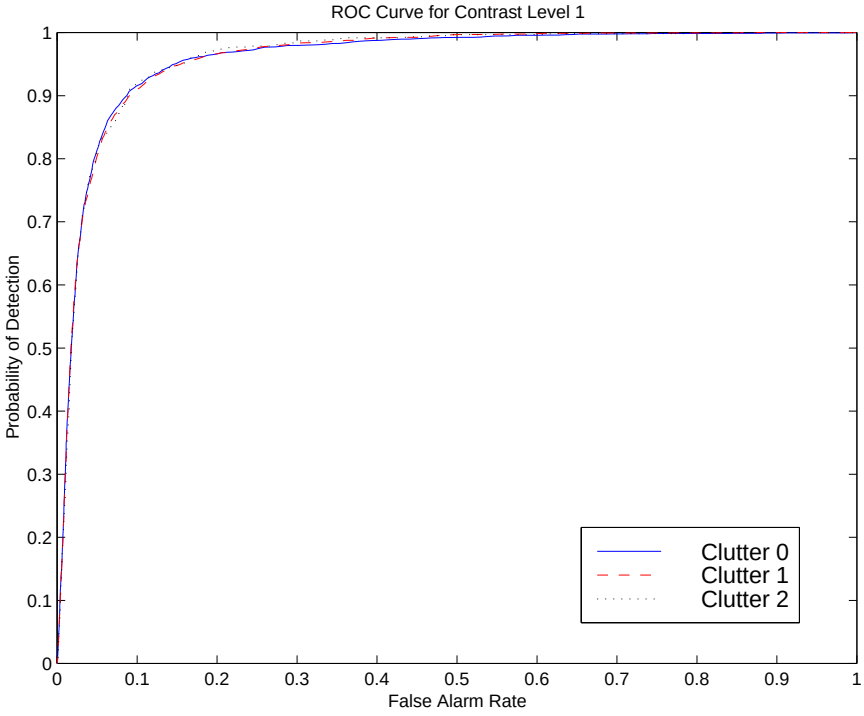
The feature selection process described earlier was run over all of the features described in section 2.3. The eight optimal features were found to be the maximum intensity, four SVD of FFT components (components $(1,5)$, $(1,6)$, $(1,7)$, $(1,8)$ and $(2,8)$), the $(1,2)$ FFT coefficient and the skewness. The performance of this combined detector is shown for a number of different target clutter/contrast regimes in Figures 2.10 and 2.11

2.5 Conclusion

A multiple feature based system for low level classification of targets detected by the prescreening algorithm has been developed. The ROC curves shown in Figure 2.10 indicate that the number of false alarms can be significantly reduced with only a marginal decrease in the detection rate. It is expected that some further performance improvement could be achieved by consideration of some more features such as output from a matched filter, linear prediction coefficients or some of the other features mentioned in the target detection report [5] and in Redding [3]. One last useful technique to consider is referred to as "bagging and boosting", where the results of one multi-feature system are passed

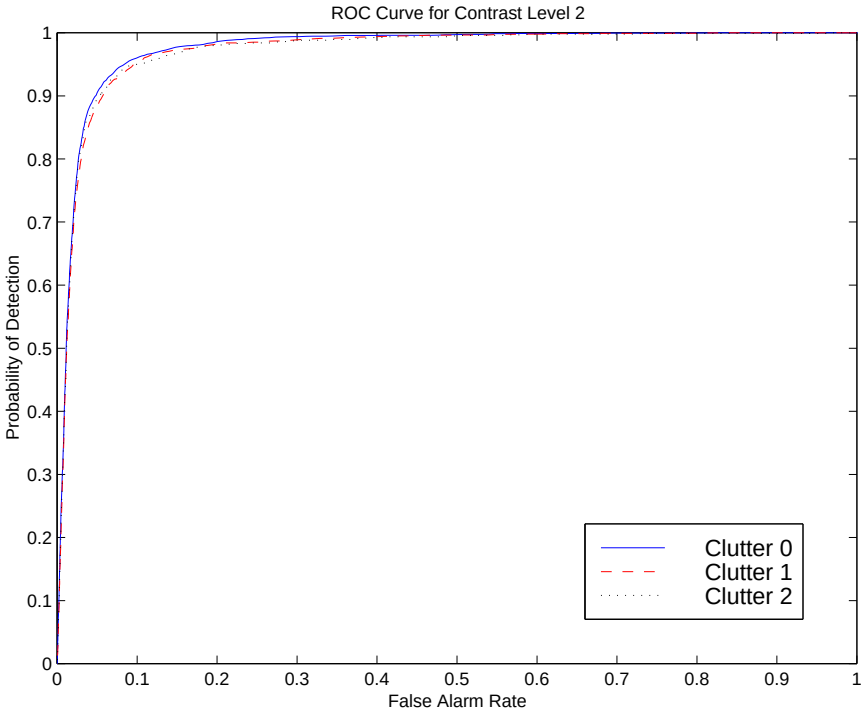


(a) Contrast Level 0

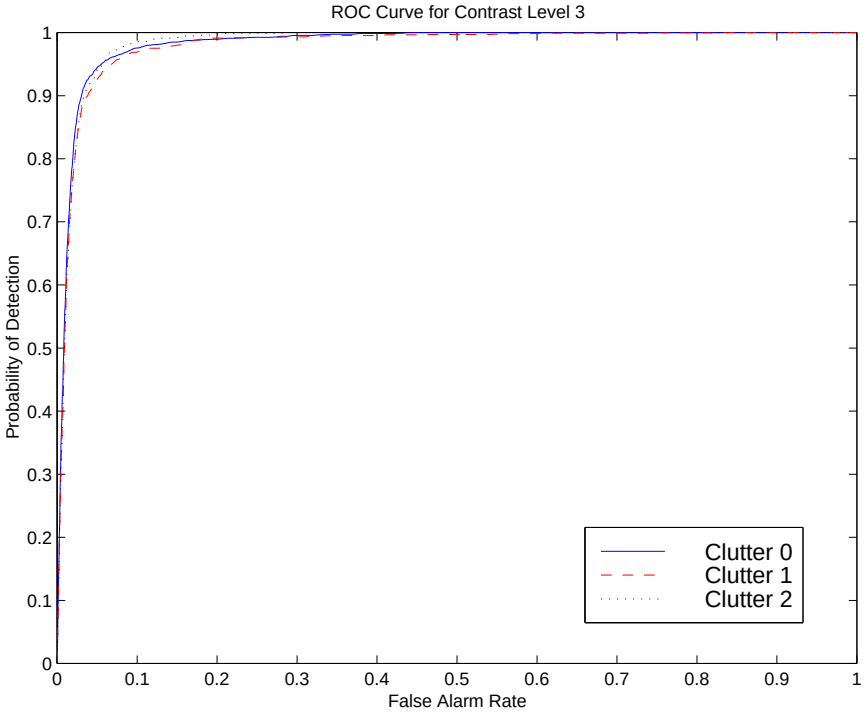


(b) Contrast Level 1

Figure 2.10: *Performance of the combined classifier for a variety of clutter/contrasts*



(a) Contrast Level 2



(b) Contrast Level 3

Figure 2.11: *Performance of the combined classifier for a variety of clutter/contrasts*

on to another. This technique can be used to build a better performing classifier from a hierarchy of lesser performing ones.

References

1. Gonzalez R.C. and Woods R.E., *Digital Image Processing* Addison-Wesley, Reading, 1992.

2. Redding N.J., Estimating the Parameters of the K-Distribution in the Intensity Domain, DSTO Draft Technical Report.

3. Redding N.J., Design of the Analysts' Detection Support System for Broad Area Aerial Surveillance, DSTO-TR-0746.

4. Fisher R.A., "The Use of multiple measurements in taxonomic problems," *Annals of Eugenics*, Vol 7, pp 179–188, 1936.

5. Schroeder J., Cooke T. and Zhang J., Target Detection Survey, CSSIP CR-11/99, May 1999.

6. Cover T.M., The best two independent measurements are not always the best, IEEE Transactions on Systems, Man and Cybernetics, Vol 4 No. 5, pp 116-117, 1974.

7. Anderson T.W. and Bahadur R.R, Classification into two multivariate normal distributions with different covariance matrices, Annals of Mathematical Statistics, Vol 33, pp 420-431, June 1962.

APPENDIX A: The Linear Discriminant

The linear discriminant presented in this appendix was derived with the help of Michael Peake of CSSIP. It provides a faster method for linear discrimination than methods like SVM, and usually produces slightly better results than the Fisher discriminant, since the separation criteria that is being optimised in this case is directly related to the PD and FAR. It is also very similar to work presented by Anderson and Bahadur [7], although they do not provide computational details for their method.

A.1 Statement of the problem

Suppose that two different classes of objects are normally distributed in a feature space of dimension M. The first class has mean μ_1 and correlation $\mathbf{C}_1$ while the second class has mean μ_2 and correlation $\mathbf{C}_2$. Further suppose that a separating hyperplane is given by the equation $\mathbf{n}.\mathbf{x} = c$. The object is then to find vector $\mathbf{n}$ and constant c to maximise the weighted correct classification rate

$$\beta \int_{\mathbf{n}.\mathbf{x}<c} P_1(\mathbf{x})dx + \int_{\mathbf{n}.\mathbf{x}>c} P_2(\mathbf{x})d\mathbf{x} - 1 \tag{1}$$

where $P_1(\mathbf{x})$ and $P_2(\mathbf{x})$ are probability distribution functions for classes 1 and 2 respectively, and are given by

$$\begin{aligned} P_1(\mathbf{x}) &= \frac{1}{\sqrt{2\pi|\mathbf{C}_1|}} \exp\left(-\frac{1}{2}(\mathbf{x}-\mu_1)^T\mathbf{C}_1^{-1}(\mathbf{x}-\mu_1)\right) \\ P_2(\mathbf{x}) &= \frac{1}{\sqrt{2\pi|\mathbf{C}_2|}} \exp\left(-\frac{1}{2}(\mathbf{x}-\mu_2)^T\mathbf{C}_2^{-1}(\mathbf{x}-\mu_2)\right) \end{aligned}$$

A.2 Derivation of the Radon Transform of a Gaussian

The Radon transform of a multi-dimensional Gaussian is required for the later work, so its derivation is presented here first. Suppose

$$I(c) = \int_{\mathbf{n}.\mathbf{x}=c} \exp\left(-\frac{1}{2}(\mathbf{x}-\mu)^T\mathbf{C}^{-1}(\mathbf{x}-\mu)\right) d\mathbf{x}$$

where $\mathbf{C}$ is a positive definite symmetric matrix. Let $\mathbf{y} = \mathbf{x} - \mu - \mathbf{w}$ where $\mathbf{w}$ is chosen to satisfy $\mathbf{n}.\mathbf{w} = c - \mathbf{n}.\mu$. Hence $\mathbf{n}.\mathbf{y} = \mathbf{n}.\mathbf{x} - \mathbf{n}.\mu - \mathbf{n}.\mathbf{w} = 0$ when $\mathbf{n}.\mathbf{x} = c$.

Substitution gives

$$\begin{aligned} I(c) &= \int_{\mathbf{n}.\mathbf{y}=0} \exp\left(-\frac{1}{2}(\mathbf{y}+\mathbf{w})^T\mathbf{C}^{-1}(\mathbf{y}+\mathbf{w})\right) d\mathbf{y} \\ &= \int_{\mathbf{n}.\mathbf{y}=0} \exp\left(-\frac{1}{2}\left[\mathbf{y}^T\mathbf{C}^{-1}\mathbf{y} + 2\mathbf{y}^T\mathbf{C}^{-1}\mathbf{w} + \mathbf{w}^T\mathbf{C}^{-1}\mathbf{w}\right]\right) d\mathbf{y}. \end{aligned}$$

Now $\mathbf{w}$ can be chosen to satisfy both the condition mentioned above (that $\mathbf{n}.\mathbf{w} = c - \mathbf{n}.\mu$) and $\mathbf{w}^T\mathbf{C}^{-1}\mathbf{y} = 0$ on $\mathbf{n}.\mathbf{y} = 0$ by setting

$$\mathbf{w} = \frac{(c - \mathbf{n}.\mu)\mathbf{C}\mathbf{n}}{\mathbf{n}^T\mathbf{C}\mathbf{n}},$$

which always exists since $\mathbf{C}$ is positive definite. Thus

$$I(c) = \exp\left(-\frac{1}{2}\frac{(c-\mathbf{n}.\mu)^2}{\mathbf{n}^T\mathbf{C}\mathbf{n}}\right) \int_{\mathbf{n}.\mathbf{y}=0} \exp\left(-\frac{1}{2}\mathbf{y}^T\mathbf{C}^{-1}\mathbf{y}\right) d\mathbf{y}. \tag{2}$$

From the original definition, $I(c)$ must satisfy

$$\int_{c=-\infty}^{\infty} I(c)dc = \sqrt{2\pi|\mathbf{C}|}$$

so substituting equation (2) into this gives

$$\int_{c=-\infty}^{\infty} \exp\left(-\frac{1}{2}\frac{(c-\mathbf{n}.\mu)^2}{\mathbf{n}^T\mathbf{C}\mathbf{n}}\right) dc \int_{\mathbf{n}.\mathbf{y}=0} \exp\left(-\frac{1}{2}\mathbf{y}^T\mathbf{C}^{-1}\mathbf{y}\right) d\mathbf{y} = \sqrt{2\pi|\mathbf{C}|}.$$

Rearranging and evaluating the integral in c produces

$$\int_{\mathbf{n}.\mathbf{y}=0} \exp\left(-\frac{1}{2}\mathbf{y}^T\mathbf{C}^{-1}\mathbf{y}\right) d\mathbf{y} = \sqrt{\frac{|C|}{\mathbf{n}^T\mathbf{C}\mathbf{n}}}.$$

Substituting back into equation (2) results in the final expression

$$I(c) = \sqrt{\frac{|\mathbf{C}|}{\mathbf{n}^T\mathbf{C}\mathbf{n}}} \exp\left(-\frac{1}{2}\frac{(c-\mathbf{n}.\mu)^2}{\mathbf{n}^T\mathbf{C}\mathbf{n}}\right). \tag{3}$$

A.3 Solution of the Problem

A.3.1 Maximum with respect to c

Since the weighted classification error given by equation (1) is to be maximised with respect to c and $\mathbf{n}$, then the associated derivatives should also be maximised. Differentiating (1) with respect to c gives

$$\beta \int_{\mathbf{n}.\mathbf{x}=c} P_1(\mathbf{x}) d\mathbf{x} - \int_{\mathbf{n}.\mathbf{x}=c} P_2(\mathbf{x}) d\mathbf{x} = 0. \tag{4}$$

The integrals in this equation correspond to the Radon transform of a multidimensional Gaussian, which was derived earlier. Using equation (3) to evaluate these integrals produces

$$\begin{aligned} \beta \frac{1}{\sqrt{2\pi|\mathbf{C}_1|}} \sqrt{\frac{|\mathbf{C}_1|}{\mathbf{n}^T\mathbf{C}_1\mathbf{n}}} \exp\left(-\frac{1}{2}\frac{(c-\mathbf{n}.\mu_1)^2}{\mathbf{n}^T\mathbf{C}_1\mathbf{n}}\right) &= \\ \frac{1}{\sqrt{2\pi|\mathbf{C}_2|}} \sqrt{\frac{|\mathbf{C}_2|}{\mathbf{n}^T\mathbf{C}_2\mathbf{n}}} \exp\left(-\frac{1}{2}\frac{(c-\mathbf{n}.\mu_2)^2}{\mathbf{n}^T\mathbf{C}_2\mathbf{n}}\right) & \end{aligned}$$

which after rearranging yields

$$\frac{(c-\mathbf{n}.\mu_1)^2}{\mathbf{n}^T\mathbf{C}_1\mathbf{n}} - \frac{(c-\mathbf{n}.\mu_2)^2}{\mathbf{n}^T\mathbf{C}_2\mathbf{n}} = \ln\left(\beta^2 \frac{\mathbf{n}^T\mathbf{C}_2\mathbf{n}}{\mathbf{n}^T\mathbf{C}_1\mathbf{n}}\right). \tag{5}$$

A.3.2 Maximum with respect to n

The weighted correct classification rate in (1) may be rewritten as

$$\beta \int_{z=-\infty}^{c} \int_{\mathbf{n}.\mathbf{x}=z} P_1(\mathbf{x}) d\mathbf{x} - \int_{z=-\infty}^{c} \int_{\mathbf{n}.\mathbf{x}=z} P_2(\mathbf{x}) d\mathbf{x},$$

and by using the value of the integral given in equation (3), may be further expanded to

$$\begin{aligned} & \frac{\beta}{\sqrt{2\pi}} \int_{z=-\infty}^{c} \frac{1}{\sqrt{\mathbf{n}^T\mathbf{C}_1\mathbf{n}}} \exp\left(-\frac{(z-\mathbf{n}.\mu_1)^2}{2\mathbf{n}^T\mathbf{C}_1\mathbf{n}}\right) dz \\ - & \frac{1}{\sqrt{2\pi}} \int_{z=-\infty}^{c} \frac{1}{\sqrt{\mathbf{n}^T\mathbf{C}_2\mathbf{n}}} \exp\left(-\frac{(z-\mathbf{n}.\mu_2)^2}{2\mathbf{n}^T\mathbf{C}_2\mathbf{n}}\right) dz. \end{aligned}$$

Using the substitution $\xi_1 = (z - \mathbf{n}.\mu_1)/\sqrt{\mathbf{n}^T\mathbf{C}_1\mathbf{n}}$ (with a similar term for ξ_2) allows the term to be maximised to be written as

$$\frac{1}{\sqrt{2\pi}}\left[\beta\int_{\xi_1=-\infty}^{v_1}\exp\left(-\frac{\xi_1^2}{2}\right)d\xi_1-\int_{\xi_2=-\infty}^{v_2}\exp\left(-\frac{\xi_2^2}{2}\right)d\xi_2\right]$$

where $v_1=(c-\mathbf{n}.\mu_1)/\sqrt{\mathbf{n}^T\mathbf{C}_1\mathbf{n}}$ and $v_2=(c-\mathbf{n}.\mu_2)/\sqrt{\mathbf{n}^T\mathbf{C}_2\mathbf{n}}$. To maximise this with respect to $\mathbf{n}$, the derivative of the expression with respect to each of the components of $\mathbf{n}$ are taken and set equal to zero. These M equations can be expressed in vector form as

$$\frac{1}{\sqrt{2\pi}}\left[\beta\exp\left(-\frac{v_1^2}{2}\right)\frac{\partial v_1}{\partial\mathbf{n}}-\exp\left(-\frac{v_2^2}{2}\right)\frac{\partial v_2}{\partial\mathbf{n}}\right]=0.$$

Dividing through by $\exp(-v_1^2/2)$, substituting the values for v_1 and v_2 and using equation (5) (which must also be true when (1) is maximised) results in

$$\beta\frac{\partial}{\partial\mathbf{n}}\left(\frac{c-\mathbf{n}.\mu_1}{\sqrt{\mathbf{n}^T\mathbf{C}_1\mathbf{n}}}\right)-\beta\sqrt{\frac{\mathbf{n}^T\mathbf{C}_2\mathbf{n}}{\mathbf{n}^T\mathbf{C}_1\mathbf{n}}}\frac{\partial}{\partial\mathbf{n}}\left(\frac{c-\mathbf{n}.\mu_2}{\sqrt{\mathbf{n}^T\mathbf{C}_2\mathbf{n}}}\right)=0.$$

After evaluating the derivative and rearranging the result, this ends with

$$\mu_1-\mu_2+\frac{(c-\mathbf{n}.\mu_1)\mathbf{C}_1\mathbf{n}}{\mathbf{n}^T\mathbf{C}_1\mathbf{n}}-\frac{(c-\mathbf{n}.\mu_2)\mathbf{C}_2\mathbf{n}}{\mathbf{n}^T\mathbf{C}_2\mathbf{n}}=0. \tag{6}$$

A.3.3 Computational Evaluation

The answer to the stated problem of finding the optimal c and $\mathbf{n}$ may be arrived at by the simultaneous solution of equations (5) and (6). These equations may be simplified somewhat by assuming that the constant β, which until now had been arbitrary, is set so that the logarithmic term of equation (5) disappears. That is

$$\beta^2=\frac{\mathbf{n}^T\mathbf{C}_1\mathbf{n}}{\mathbf{n}^T\mathbf{C}_2\mathbf{n}}. \tag{7}$$

From this assumption, equation (5) becomes

$$\mathbf{n}^T\mathbf{C}_1\mathbf{n}(c-\mathbf{n}.\mu_2)^2=\mathbf{n}^T\mathbf{C}_2\mathbf{n}(c-\mathbf{n}.\mu_1)^2.$$

Since any discriminating hyperplane should separate the means of the two distributions, then $(c-\mathbf{n}.\mu_1)$ and $(c-\mathbf{n}.\mu_2)$ should have opposite signs. Hence rearrangement of the above equation will give

$$c=\frac{\sqrt{\mathbf{n}^T\mathbf{C}_1\mathbf{n}}\;\mathbf{n}.\mu_2+\sqrt{\mathbf{n}^T\mathbf{C}_2\mathbf{n}}\;\mathbf{n}.\mu_1}{\sqrt{\mathbf{n}^T\mathbf{C}_1\mathbf{n}}+\sqrt{\mathbf{n}^T\mathbf{C}_2\mathbf{n}}}.$$

Substituting this expression into equation (6), using the fact that **n**s magnitude may be scaled arbitrarily (since the equations have this as a degree of freedom) and rearranging gives

$$\mathbf{n} = (\mathbf{C}_1 + \beta\mathbf{C}_2)^{-1}(\mu_2 - \mu_1). \tag{8}$$

Lemma: If $\mathbf{C}_1$ and $\mathbf{C}_2$ are positive definite symmetric matrices, then the set of equations (7) and (8) has a solution which exists and is unique .

Proof: Firstly it is noted that if $\mathbf{A}$ and $\mathbf{B}$ are positive-definite, real symmetric matrices, then so are $\mathbf{AB}$, $\mathbf{A}^{-1}$ and there exists some matrix $\mathbf{C} = \mathbf{A}^{1/2}$ satisfying $\mathbf{C}^2 = \mathbf{A} = \mathbf{C}\mathbf{C}^T$. Furthermore, any positive-definite real symmetric matrix has a complete set of eigenvectors, all with positive eigenvalues.

Equation 7 may be written (after setting $\mu = \mu_2 - \mu_1$ for convenience) as

$$\beta^2 = \frac{\mu^T(\mathbf{C}_1 + \beta\mathbf{C}_2)^{-1}\mathbf{C}_1(\mathbf{C}_1 + \beta\mathbf{C}_2)^{-1}\mu}{\mu^T(\mathbf{C}_1 + \beta\mathbf{C}_2)^{-1}\mathbf{C}_2(\mathbf{C}_1 + \beta\mathbf{C}_2)^{-1}\mu}.$$

Since $\mathbf{C}_1$ is positive definite, then rearranging and making the substitutions

$$\begin{aligned} \nu &= \mathbf{C}_1^{-1/2}\mu \qquad \text{and} \\ R &= \mathbf{C}_1^{-1/2}\mathbf{C}_2\mathbf{C}_1^{-1/2}, \end{aligned}$$

where $\mathbf{R}$ must be symmetric and positive definite, simplifies the previous equation to

$$\beta^2 = \frac{\nu^T(\mathbf{I} + \beta\mathbf{R})^{-2}\nu}{\nu^T(\mathbf{I} + \beta\mathbf{R})^{-1}\mathbf{R}(\mathbf{I} + \beta\mathbf{R})^{-1}\nu}.$$

The right-hand side of this may be written in terms of the eigenvectors and eigenvalues of $\mathbf{R}$. In particular, let $\rho_\mathbf{i}$ be the eigenvectors and λ_i the eigenvalues of $\mathbf{R}$ and let $\nu = \sum_i n_i\rho_\mathbf{i}$. Hence

$$\beta^2 = \frac{\sum_i n_i^2/(1+\beta\lambda_i)^2}{\sum_i n_i^2\lambda_i/(1+\beta\lambda_i)^2}$$

or

$$\sum_i \frac{n_i^2\lambda_i\beta^2}{(1+\beta\lambda_i)^2} = \sum_i \frac{n_i^2}{(1+\beta\lambda_i)^2}. \tag{9}$$

The left-hand side of this last equation is an increasing function of β, while the right-hand side is a decreasing function of β. When $\beta = 0$, the right-hand side is greater, but the left-hand side is greater when β is large. There are no singularities in either side when β is positive. Therefore, there is exactly one positive solution for β.

QED

The proof of the lemma also provides an iterative method for the calculation of β which will always converge. Since equation (9) consists of an increasing right hand side and a decreasing left hand side, then a binary search on β in the equation $\beta^2(\mathbf{n}(\beta)^T\mathbf{C}_2\mathbf{n}(\beta)) = (\mathbf{n}(\beta)^T\mathbf{C}_1\mathbf{n}(\beta))$ will always produce the correct solution for β. $\mathbf{n}$ and c may be obtained by substituting back into equations 8 and 6 respectively.

A.4 Conclusion

A method has been developed for maximising the weighted classification rate $\beta PD - FAR$ for a linear discriminant which separates two multi-dimensional Gaussian distributed classes. The computational procedure described only produces results for one specific value of β but if a stable method for solving equations (5) and (6) can be found, this would allow β to be specified for the required application. The procedure is much faster than many methods such as the SVM, and unlike the Fisher discriminant (which for the special case of $\beta = 1$ is the same as the discriminant discussed here) it maximises a quantity which is physically related to ROC curves. In most cases, the Fisher discriminant seems to give very similar results to this linear discriminant, but in certain cases the procedure described here yields a distinct advantage in determining the optimal ROC curve.

Chapter 3

Feature Extraction II

3.1 Introduction

In the previous report [2], a number of target features had been proposed, which when used in combination with a linear discriminant produced a fairly high performance low-level classifier. While this classifier reduced the number of false alarms by a factor of fifteen to twenty, this performance can still be improved.

The following chapters examine a number of topics including previously unconsidered features, the effect of speckle filtering on low level classification performance and adaptive detection. Finally, the best overall performance results are presented, which were obtained using a linear discriminant with 9 adaptively modified features.

3.2 The Data Set and Feature Selection

All of the results for the features outlined in this report were obtained by testing on a data set containing 22084 targets and 53961 background samples. The data set was obtained by inserting 512 actual targets into 5 large magnitude only SAR images by an insertion procedure described in Redding [1]. Then an existing prescreening algorithm was run over the images. The Regions of Interest (ROIs) picked up by this prescreener were then used to construct the data set. Each ROI was labeled as a 'hit' or a detection if the prescreening algorithm detected an inserted target, and a false alarm (or background) otherwise. Since ground-truth is not known, some of the non-inserted detections may actually correspond to true targets. Since this should correspond to a very small percentage of the total, there should be little loss in accuracy by considering these detections as false alarms. After extraction, each 64×64 ROI was sorted into groups of similar contrast and clutter measures.

Also, throughout the paper, a subset of "best" features are selected from large groups of individual features. Due to computational considerations, the way in which this is done was sub-optimal, however it is more extensive than the standard sub-optimal approach. This method, which is the same as used in the previous report, is described as follows:

1 **Initialization:** Set a feature set F equal to the empty set. At the end of the algorithm, F will be the set of "best" features for discrimination between target and background.

2 **Loop:** The following steps are repeated.

A For each possible combination of two features, repeat the following

a Let G be the union of the two features and the set F. Then calculate the discrimination performance of the set of features in G and store the results.

B The combination of two features which gave the best discrimination is then considered. If the improvement to the discrimination produced by this combination is not significant, then the loop stops. Otherwise, of these two features, the one which produced the best average discrimination in combination with all of the other features is selected and added to the set F.

The set of accepted features F produced by this algorithm can then be used to produce a detector having almost the best possible performance.

3.3 Speckle Filtering

One of the properties of SAR imagery which distinguishes it from most other types of imagery is speckle, which is a type of multiplicative noise that often severely degrades the visibility in an image. Applying any of a large number of speckle filters to the image will usually result in a much clearer picture for human analysts and for targets to appear much more obviously. In order to test whether the speckle filter could produce a similar improvement in the automated detection capability, the particular speckle filter derived in [4] was considered.

Since the speckle processing step is quite computationally intensive, only the subset of the data set corresponding to the targets and backgrounds having the lowest contrast and clutter was used for testing. For each of the images in this subset, the 9 best features from the previous report [2] were calculated and then used to construct a ROC curve. Then, for the same set of images, the same 9 features were calculated after a speckle filter had been applied to each image. This was repeated for a number of speckle filter parameters, and the effect on the ROC curve is shown in Figure 3.1. The speckle filter parameter seems to have little effect on the overall performance.

There are a number of reasons why the effect of the speckle filtering may not be quite as bad as shown. Firstly, the 9 features used in this comparison had been chosen specifically to optimise the discrimination between speckled images. This means that it is possible for a different set of features to be selected which optimise the despeckled image discrimination and produce somewhat better results than those shown. It is also possible that a different type of speckle filter could give improved results. Even keeping the above explanations in mind however, the results of the comparison seems to indicate that speckle filtering makes computational target detection significantly more difficult, and for this reason will not be considered in the remainder of the report.

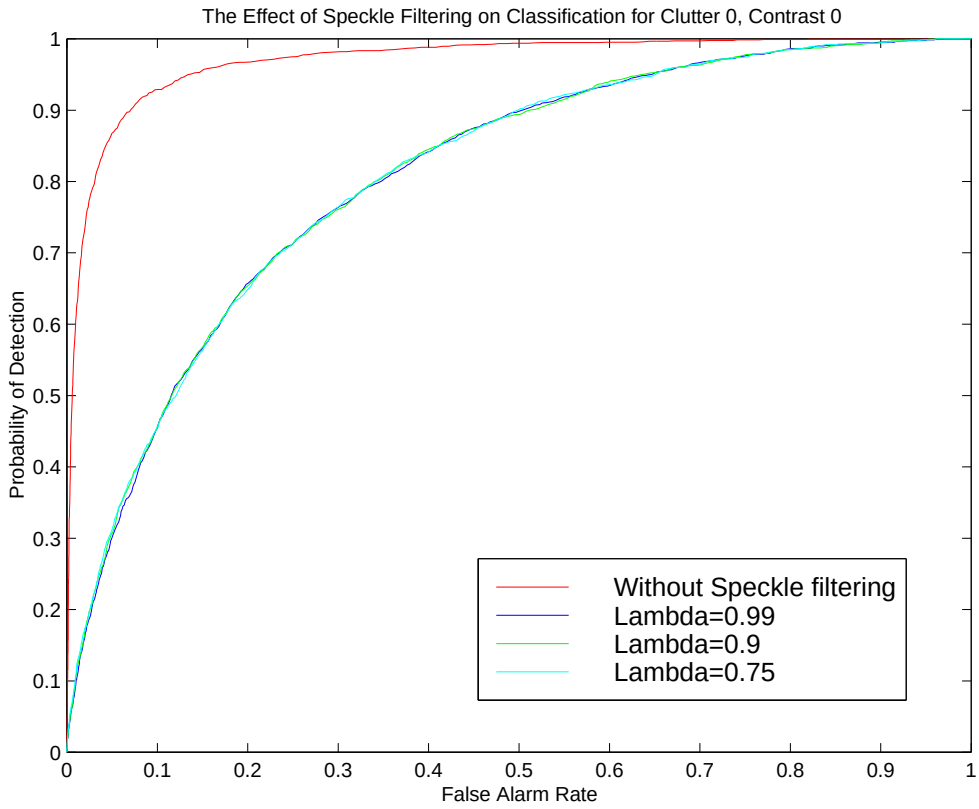


Figure 3.1: *Comparison of ROC curves before and after speckle filtering*

3.4 Individual Target/Background Features

This section examines a large number of individual features which could be useful for target/background discrimination. Many of these features may not be especially useful by themselves, but in combination with the other features listed here and in the previous report [2], could significantly improve the system performance. Some possible methods for combining these features are described in the next section.

3.4.1 Singular Value Decompositions

Any rectangular image can be expressed as a matrix whose elements correspond to the grey-scale intensities of the image. The singular value decomposition (SVD) of this matrix $\mathbf{X}$ can be uniquely written as

$$\mathbf{X} = \mathbf{U}\mathbf{D}\mathbf{V}^T$$

where $\mathbf{D}$ is a diagonal matrix and $\mathbf{U}$ and $\mathbf{V}$ are unitary matrices. Previously in [2], it was reported that both the SVD of the 2D Fourier transform, and the first diagonal entry of $\mathbf{D}$

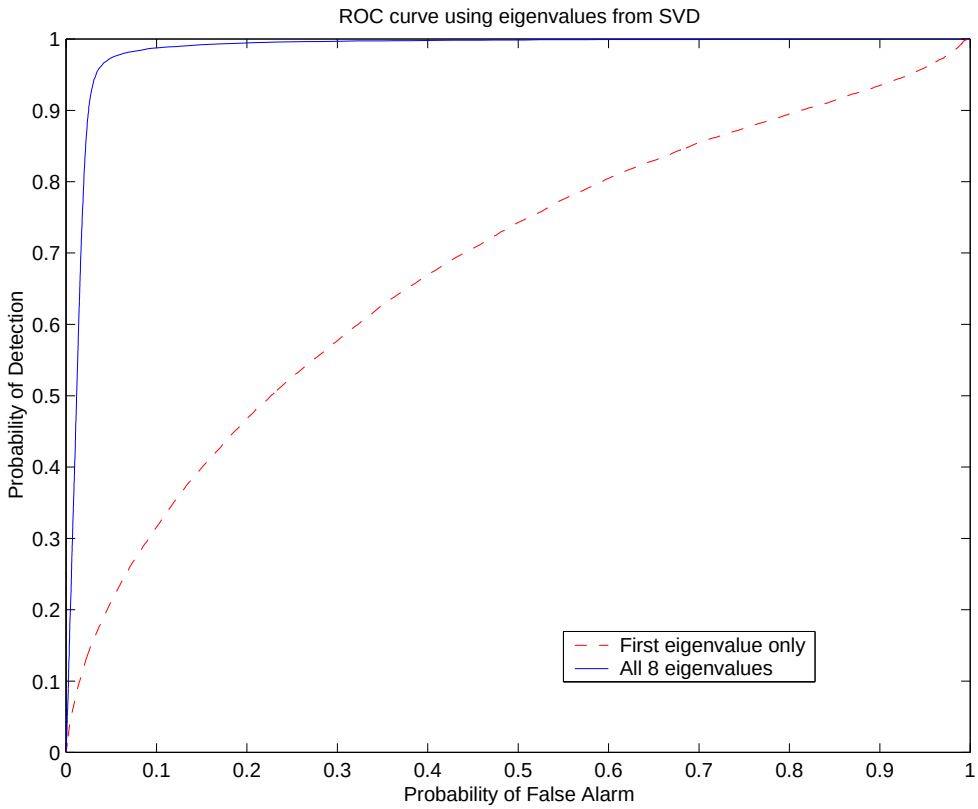
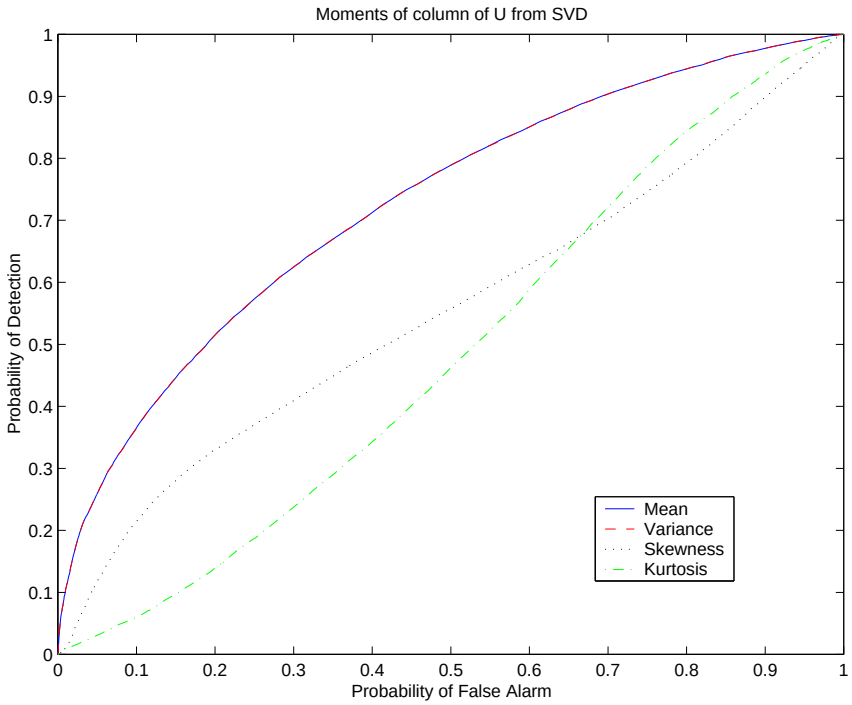


Figure 3.2: *ROC curves obtained using eigenvalues from the SVD*

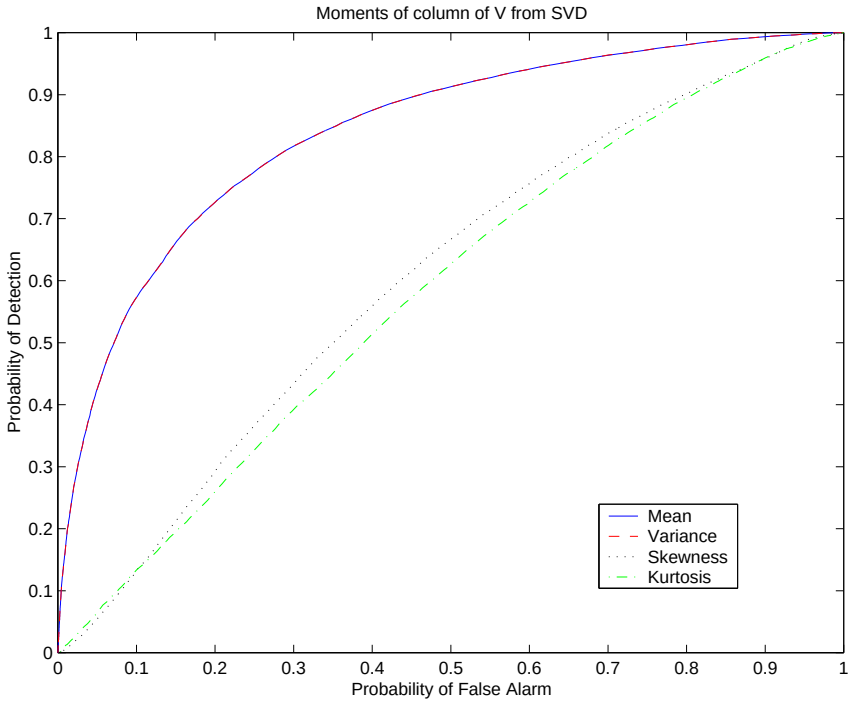
provided good target/background discrimination information. While the first eigenvalue corresponded to the majority of the target's energy, it appears that for discrimination between a target and a bright background spot that the other lower energy eigenvalues play an especially important role. Figure 3.2 shows the improvement in detection obtained by using all 8 eigenvalues from the SVD. For this imagery, the great majority of the information appears to be captured by the seventh eigenvalue [1]. Much of this information is duplicated however, so that there is not too great a decrease in performance by considering all singular values except for the seventh. It also appears that a significant performance improvement can be obtained by using a 9×9 image instead of an 8×8, although the reason for this is not clear. A more detailed discussion of this feature is given in [10].

The first column of the matrices $\mathbf{U}$ and $\mathbf{V}$ also provide useful discriminating ability, but nowhere near to the same extent as the eigenvalues. The majority of the first eigenvector information appears to be found in the moments of these columns, and ROC curves for the first 4 moments of $\mathbf{U}$ and $\mathbf{V}$ are shown in Figure 3.3.

[1] It was found in a later report that this feature only worked with the inserted targets, and in fact detected the linear smoothing applied to the edges of the target after insertion. This feature had poor performance on non-simulated data sets



(a) Moments of **U**



(b) Moments of **V**

Figure 3.3: *ROC curves for the moments of the first columns of the singular value decomposition*

3.4.2 Rank Order Statistics

Rank order statistics of an image are obtained by ordering all of the pixels of the image by their brightness. Each position in this sorted list corresponds to an individual statistic. For instance, a point chosen at the end of the list would correspond to the brightest or the dimmest point in the image, while a point chosen from the exact center would correspond to the median. In the previous report [2], only the maximum intensity was considered, which was found to be an extremely useful discriminant. A number of papers including [5] and [6] indicate that other features based on rank order statistics could also be useful.

In [5], only the rank order statistics of the original image are considered. As a result, an optimum subset of all 64 statistics for an 8×8 image was computed, using the method described in Section 2. The ROC curve generated by the best 4 rank order statistics is displayed in Figure 3.4.

Billard et. al. [6] however uses two non-parametric hypothesis tests to test whether the pixels are identically distributed. The first test discussed was the Wilcoxon test, which is used to determine if two distributions have the same median. The second hypothesis test, the Mann-Whitney test, requires two sets of observations, which is not applicable in this instance.

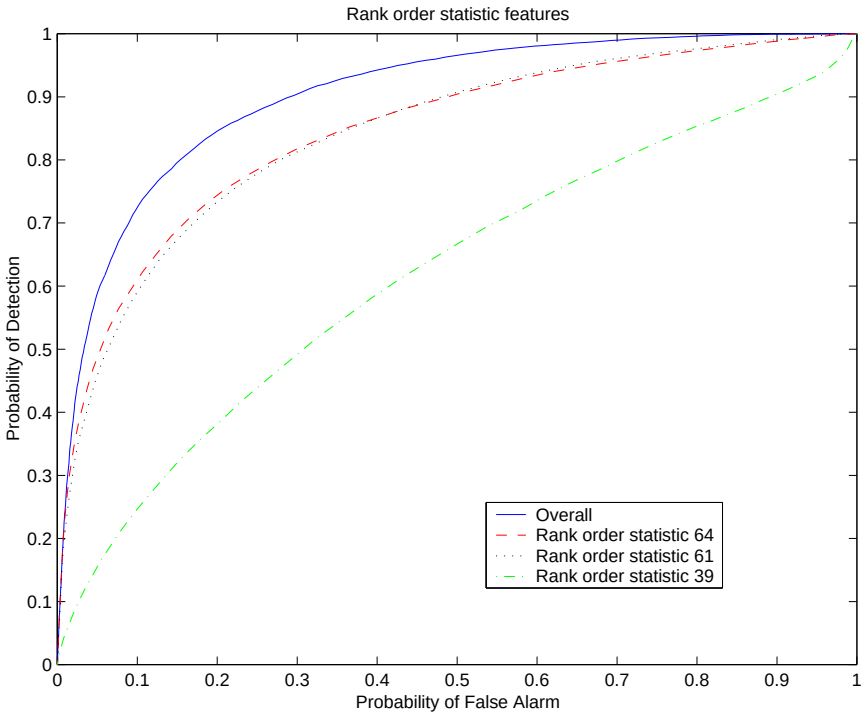
The Wilcoxon test may be used to test whether a distribution x is symmetric about its mean by testing whether $x-\mu$ and $\mu-x$ have the same median. One method for using this test to distinguish target from background could be to just use the Wilcoxon test statistic of the individual pixel intensities. This would be a measure of the distribution's skewness, which is probably higher for an image containing a target, and should similar detection capabilities to the statistical third order moment discussed in the previous report [2]. The method in [6] however applies the Wilcoxon test to the Walsh transform (minus the first row) of the pixel distribution. The Walsh transform of any vector of identically and independently distributed (i.i.d.) random variables should produce a symmetrical set of i.i.d variables as an output. A target however will not be generated by the same distribution as background, and so should show up as an asymmetry in the random variables generated by the Walsh transform. This asymmetry will be captured by the Wilcoxon test statistic.

Using both of the above Wilcoxon test statistic based features in a detection algorithm produced the ROC curve shown in Figure 3.4.

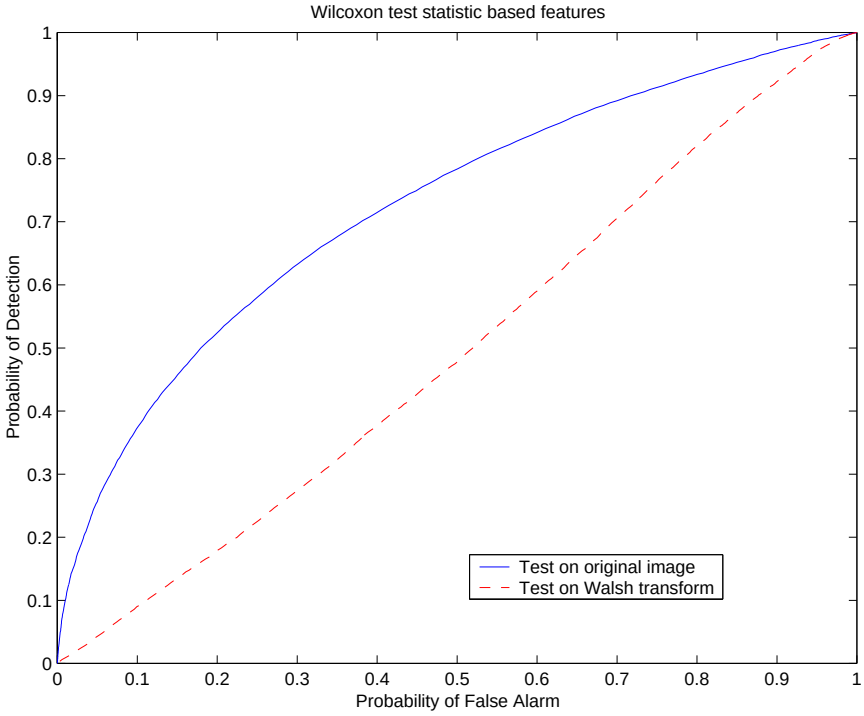
3.4.3 Fractional Brownian Motions

Guillemet et. al. [7] models images containing background clutter in terms of a fractal like stochastic process known as a noisy fractional Brownian motion (FBM). An FBM is a stochastic Gaussian process $B_H(t)$ with increments which are stationary and self-similar on a scale related to the fraction H, which is a real number between 0 and 1 inclusive. Guillemet et. al. derives a set of quantities related to Figure 3.5 to determine the parameters of a noisy FBM.

Assuming that the image is an instance of a noisy FBM, then for a particular inter-pixel distance d, the random variable defined by $\epsilon = E - (A + B + C + D)/4$ (where



(a) The three "best" rank statistics



(b) Wilcoxon test statistic based features

Figure 3.4: *ROC curves for rank order statistics based methods*

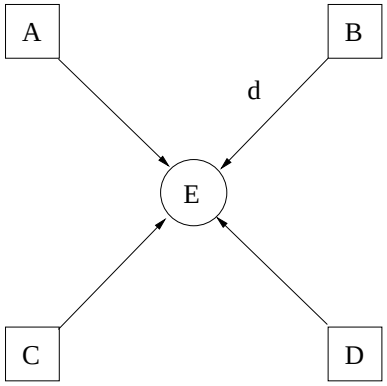


Figure 3.5: *Geometry used for determining parameters of an FBM*

A, B, C, D and E are the intensity values of the pixels shown in Figure 3.5) should be normally distributed with mean 0 and variance

$$E(\epsilon^2) = \sigma^2 d^{2H}(1 - 2^{H-2} - 2^{2H-3}) + 2\Sigma^2.$$

Here H is the fractional parameter of the FBM, σ^2 is its variance and Σ^2 is the variance of the noise. By estimating $E(\epsilon^2)$ for a number of inter-pixel sizes, a curve can be fitted and hence the noisy FBM parameters can be calculated.

A few changes were made to the way in which these parameters were used for the purposes of this report. Guillemet et. al. only used a small subset of possible blocks for calculating $E(\epsilon^2)$ as a function of d, whereas for this test all possible blocks for a particular d were used. Also, since the noisy FBM parameters will be continuous functions of the individual $E(\epsilon^2)$s, to avoid extra computation the computed values of $E(\epsilon^2)$ for each d were used as features instead of the parameters. The ROC curves for these features for a variety of block sizes are shown in Figure 3.6.

3.4.4 Lincoln Labs Features

Under the terms of the Strategic Target Algorithm Research (STAR) contract between the Advanced Research Projects Agency (ARPA), the United States Air Force, and a number of laboratories (Environmental Research Institute of Michigan (ERIM), Rockwell International Corporation and Loral Defense Systems), a number of features for target/background discrimination were developed for polarimetric 0.3m SAR. The most promising of these features were combined with features developed at Lincoln Labs to produce a suite of fifteen features for testing on SAR imagery. The results of tests performed on these fifteen features were published by Kreithen et. al. [8], who found that the fractal dimension of the image was the best performing of the features which they considered.

The fractal or Hausdorff dimension of a continuous shape can be calculated by considering the minimum number N_d of squares having sides of length d necessary to completely

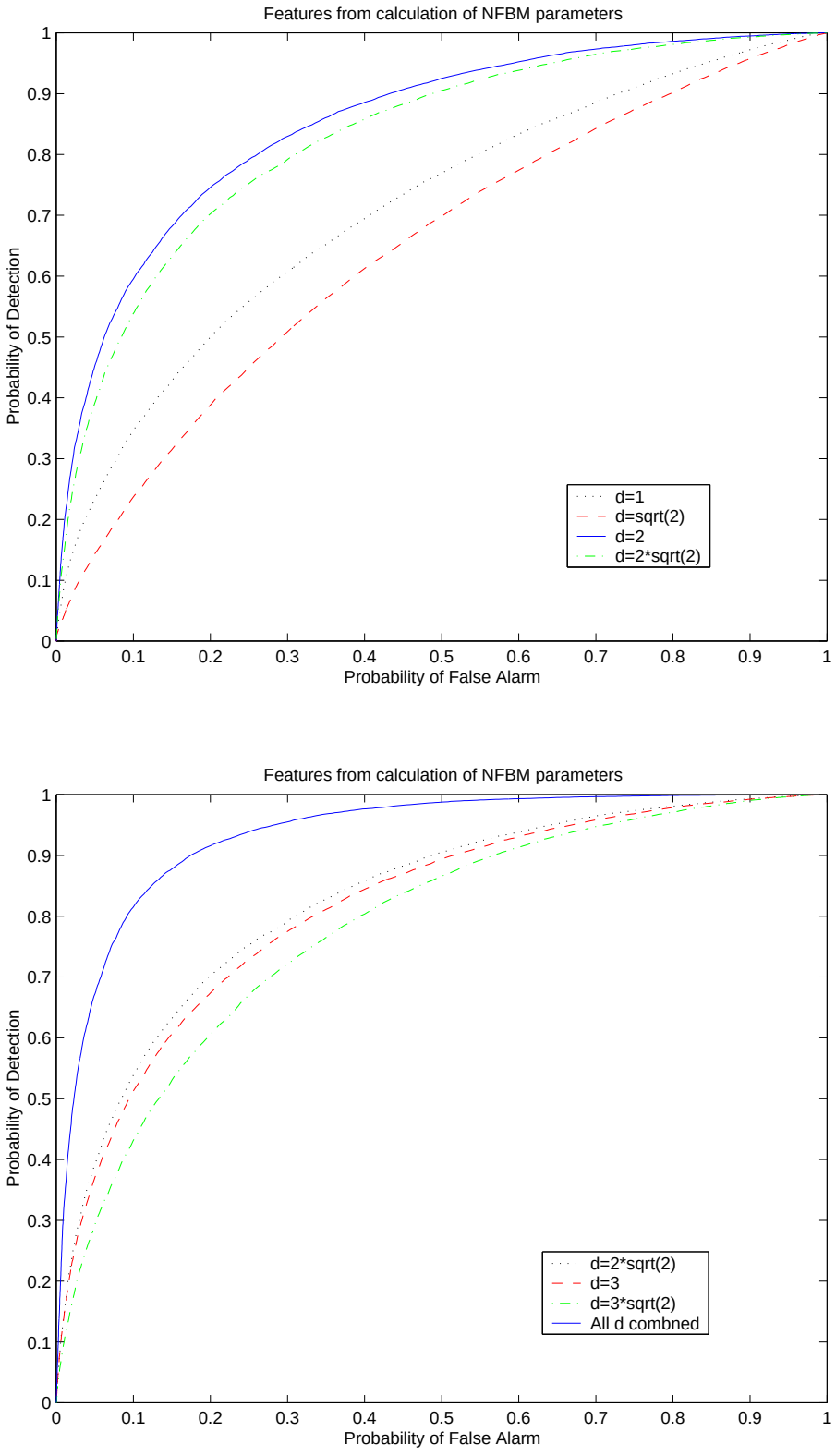


Figure 3.6: *Noisy fractional Brownian motion features for various block sizes*

cover the shape. As d becomes very small, a plot of $\log(N_d)$ against $\log(d)$ should become roughly linear, and the slope of this line will be the negative of the fractal dimension. In the digital case, d is restricted to integer values and so the best estimate for the fractal dimension of a binary shape will be to calculate

$$D = \frac{\log(N_1) - \log(N_2)}{\log(2) - \log(1)}.$$

To calculate a measure for the fractal dimension of SAR imagery, it is first necessary to reduce the grey-scale image to a shape. This may be done by a rank ordered thresholding, so that the brightest M pixels from the image are set to one, while the remaining pixels are set to zero. The fractal dimension of the shape represented by the ones may then be calculated. A target would be expected to correspond to a compact set of bright points, corresponding to a fractal dimension of about 2, whereas the bright points from background should be more dispersed, corresponding to a lower fractal dimension.

Figure 3.7 shows a ROC curve displaying the ability of the fractal dimension to discriminate targets from background for the imagery in this project. From this figure, the fractal dimensions for both 8×8 and 10×10 images appear to work very poorly. This result is strengthened by remarks in Kreithen et. al. [8], which state that although the Lincoln Labs features perform well for imagery obtained using a polarimetric whitening

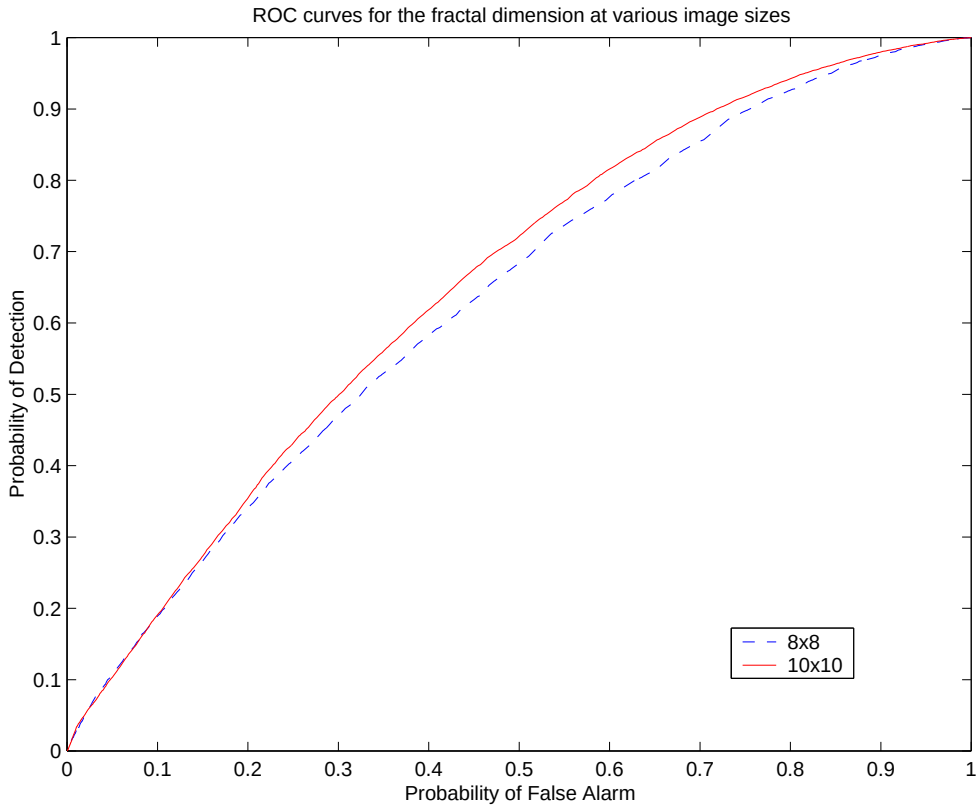


Figure 3.7: *ROC curves obtained using the fractal dimension for variously sized imagery*

filter (PWF), they perform little better than chance on 1m H-H polarization data, which is similar to the imagery used in JP129.

Due to the poor performance of this feature, which gave the best results for the Lincoln Labs data, the testing of the remaining features in this test suite will be postponed in favour of more likely candidate features.

3.4.5 Average edge intensities

One of the properties which should distinguish target containing backgrounds from normal background is the presence of edges. If an image did correspond to a target, then each row/column of the image passing through the target should contain two edges, one having increasing intensity while the other having decreasing intensity. The average of the two edges for each row/column could then be used as a feature in a classifier, and the results obtained by doing this are shown in Figure 3.8. To obtain these curves, the edge intensities were calculated using a simple two point ratio edge detector on rows and columns through the brightest spots on the possible targets.

The results from the test show that edges in the azimuthal direction provide a much greater discrimination ability, probably due to the higher correlation between pixels in the range direction. In fact, when both features were used together in a classifier, the range direction edge information did not contribute significantly to the overall detector performance.

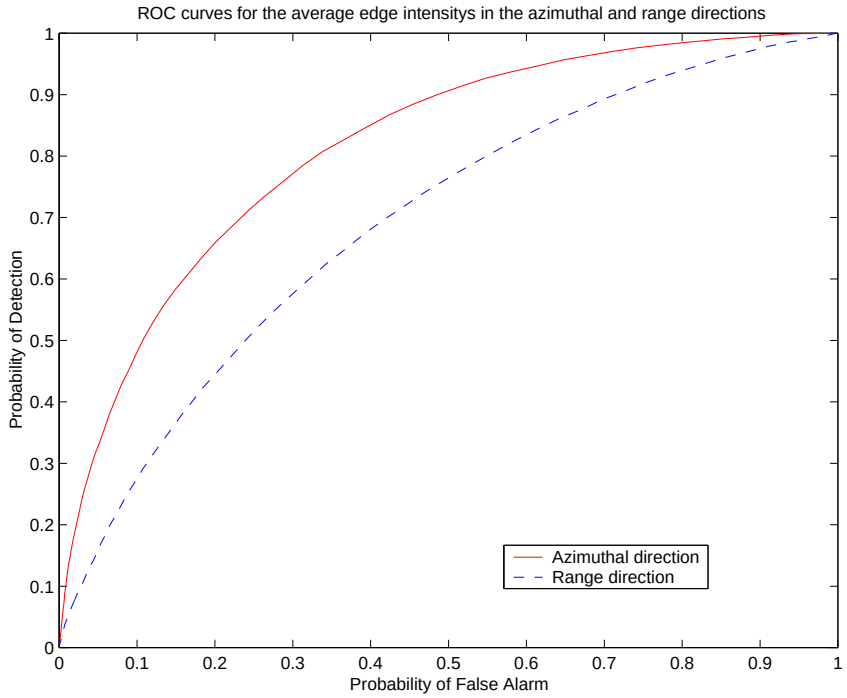


Figure 3.8: *ROC curves obtained from the average edge intensity*

3.4.6 Co-occurrence matrix features

Features such as K-distribution parameter estimates and image histogram moments have already been considered. These referred to as single point statistics since they consider only the distributions of individual points without considering the effects of correlation. The co-occurrence matrix is an example of a two point statistic, which is often used to great effect in texture classification.

A co-occurrence matrix can be calculated by considering all pairs of image pixels separated by some fixed distance and direction. For an image with M gray levels, the co-occurrence matrix will have size $M \times M$, and can be calculated in the following way.

- Initialise the matrix $\mathbf{C}$ to zero.
- For each pair of pixels separated by the correct distance and direction, do the following.
  1 Set i and j to be the gray level intensities of the first and second pixels respectively.
  2 Add 1 to the element in the ith row and jth column of the matrix $\mathbf{C}$.
- Divide $\mathbf{C}$ by the total number of pixel pairs used. This matrix $\mathbf{P}$ is now a co-occurrence matrix of the original image.

There are obvious similarities between a co-occurrence matrix and its single point analogue, the histogram. Unlike the histogram though, different co-occurrence matrices may be calculated for the image by using other interpixel separations and directions. Due to the fast decrease of correlation with distance in this imagery, results have only been calculated for neighbouring pairs of pixels in the range and the azimuthal directions.

For target detection purposes, the co-occurrence matrices are only calculated for 8×8 sized images, so the grey levels have been transformed to take only 8 different values instead of the original 256. This requantization prevents the co-occurrence matrices from being too sparse. Once the co-occurrence matrices have been calculated, features can then be obtained from them, to be used in classification. The co-occurrence matrix based features tested are listed below.

$$\textbf{Entropy} = \sum_{i=1}^{M}\sum_{j=1}^{M} P_{ij} \log P_{ij}$$

$$\textbf{Contrast} = \sum_{i=1}^{M}\sum_{j=1}^{M} (i-j)^2 P_{ij}$$

$$\textbf{Energy} = \sum_{i=1}^{M}\sum_{j=1}^{M} P_{ij}^2$$

$$\textbf{Maximum Probability} = \overset{max}{i,j}\ P_{ij}$$

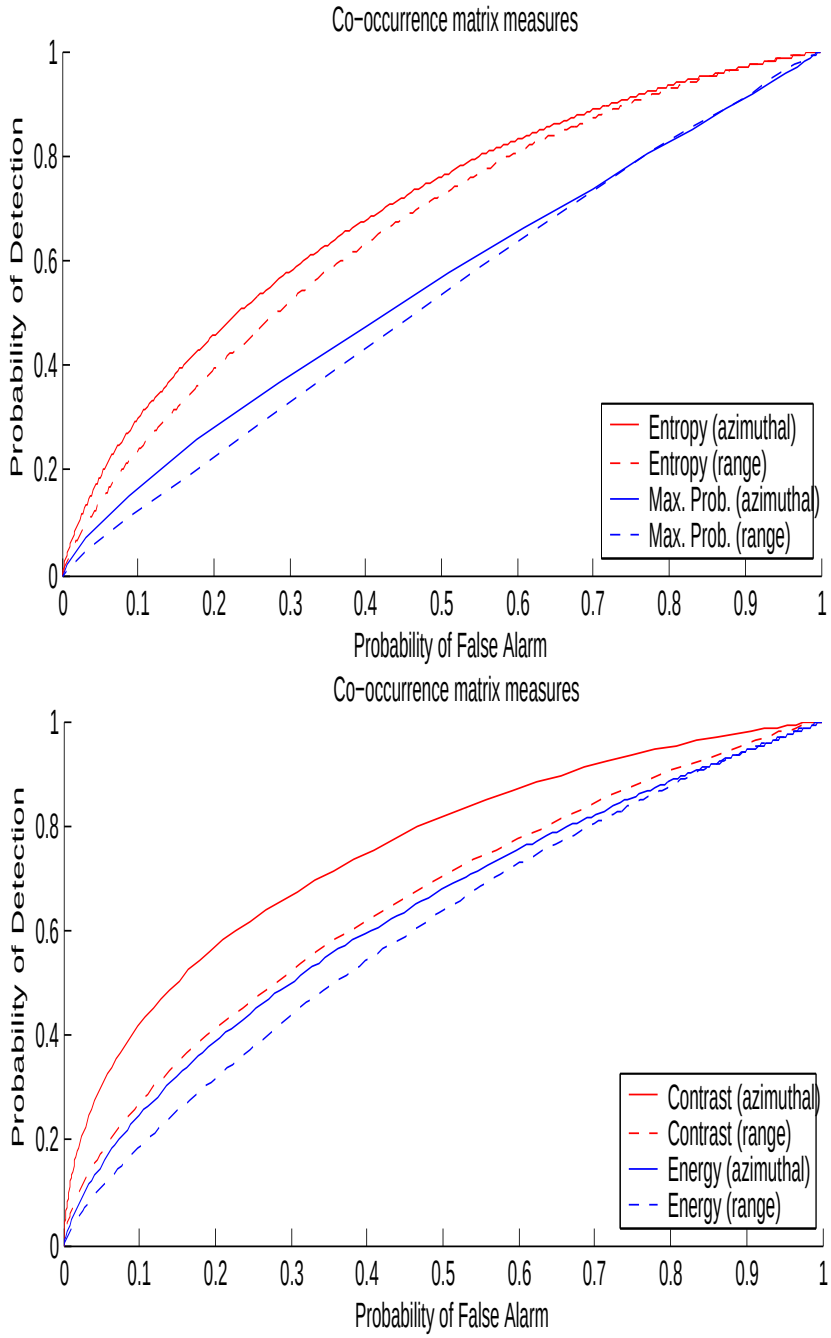


Figure 3.9: *Detection characteristics of co-occurrence matrix features*

As with the edge detection feature discussed previously, the ROC curves in Figure 3.9 show a significantly better performance in the azimuthal direction than in the range direction.

3.4.7 Other features

In the target detection survey [9] and other reports (such as [1]), numerous possible features have been mentioned. While many of these have been extensively tested in the previous [2] and current reports, there are still numerous possibilities that could be explored. Of the features not yet considered in detail, Redding [1] for instance cites multi-point statistics, Gabor filters, wavelets and multi-scale autoregressive (MAR) models as possible features. While some testing has been carried out with MAR on a limited data set as described in the target detection survey [9], the rest of these features have not been considered for the moment, for a variety of reasons.

In the examined SAR imagery, there is an extremely rapid decrease in the correlation with separation length. Also, previous tests on two-point statistics and edge intensity statistics indicate that all of the useful correlation information is in the azimuthal direction of the imagery. These two facts imply that multi-point statistics will add little information to that already obtained by the co-occurrence matrix measure. There would also be problems with the sparsity of the multi-point histogram, which could be overcome for the co-occurrence matrix measures by requantising the image. This technique however could not be used to reduce the sparsity by the same amount for multi-point histograms, so this would decrease the feature's effectiveness.

Most of the features considered so far have been applied to 8×8 sized images. The Gabor filter is unsuited to such small image sizes. In fact, for these small images, the Gabor transform is equivalent to the Fourier transform of a windowed version of the image. The Fourier transform has already been considered in the first prescreening report, so it is not expected that the Gabor filter over the small window will contribute any extra information. For much larger window sizes, the Gabor filter is more expensive to calculate and looks to behave similarly to an adaptive Fourier transform feature, which is explained in section 3.6. The Gabor filter also requires the correct choice of windowing function, which may only be determined by trial and error.

Wavelet expansions effectively expand an image as a sum of basis functions which satisfy self-similarity properties. The Fourier series expansion already considered in [2] is a special case of this type of formulation, but there are tens of other possible wavelets, including Haar, Daubechies, Morlet, Meyer and mexican hat wavelets. A thorough investigation of wavelets would require consideration of many of these basis functions, many of which would reproduce the same information uncovered by the self-similar features (such as the Fourier transform and fractal measures) already considered.

The target detection survey [9] also lists two other main features not yet considered fully in this report. Many of the singular value based methods listed are described in some depth in [10]. The majority of the MAR features however require complex imagery to construct multiscale pyramids and so could not be tested fully given the limited availability of complex data.

3.5 Combining Features

The selection procedure described in section 3.2 along with the recursive Fisher linear discriminant, as described in the discriminant report [3], was applied to all of the features considered in this report. This resulted in a group of the 9 'best' features to be selected. Ranked by order of usefulness, these features were

- Log of singular value #8 of a 9×9 sized image chip.
- Maximum intensity.
- The first value of the first row of U from the SVD of the FFT of the image.
- Log of the eighth value of the first row of U from the SVD of the FFT of the image.
- Fractional Brownian Motion (FBM) parameter for $d = \sqrt{2}$.
- FBM parameter for $d = 3$.
- Singular value #4 of a 9×9 sized image chip.
- FBM parameter for $d = 2$.
- Wilcoxon test statistic of original image.

In some of these cases, the logarithm of the feature was used for selection instead of the original feature, so that the distributions would appear more Gaussian (which appears to often result in a better classifier output). Using these features with the recursive Fisher discriminant gave the output results for the low level classification stage at various clutter and contrast regimes shown in Figures 3.10 and 3.11. It should be stressed however that this result has only been tested robustly on inserted targets from a single image (and hence for only one resolution and set of weather conditions).

These results for the test image from the Kangaroo 95 trial corresponded to a false alarm rate of $1.2/km^2$ for a low level classification PD of 90 percent. Even better results could be obtained through the use of a non-linear discriminant such as a support vector machine. In fact, a FAR of $0.2/km^2$ for the same overall PD was indicated after the LLC stage was split into two separate stages having a PD of 94.9 percent each (where the first feature by itself was used for the first stage, and the remaining features used for the second stage).

Again, due to the limited amount of data in the current set, this result for cascading classifiers needs more testing on a wider variety of data. The tests performed so far however suggest that by splitting the LLC stage into two, there are both possible improvements in FAR and significant reductions in computational requirements. The decrease in computations is possible because the remaining eight features are not necessary to be calculated for the chips rejected by the first stage of the LLC. Alternatively, further improvements to FAR could be achieved by allowing more computationally intensive features to be computed in the second stage of the low level classification.

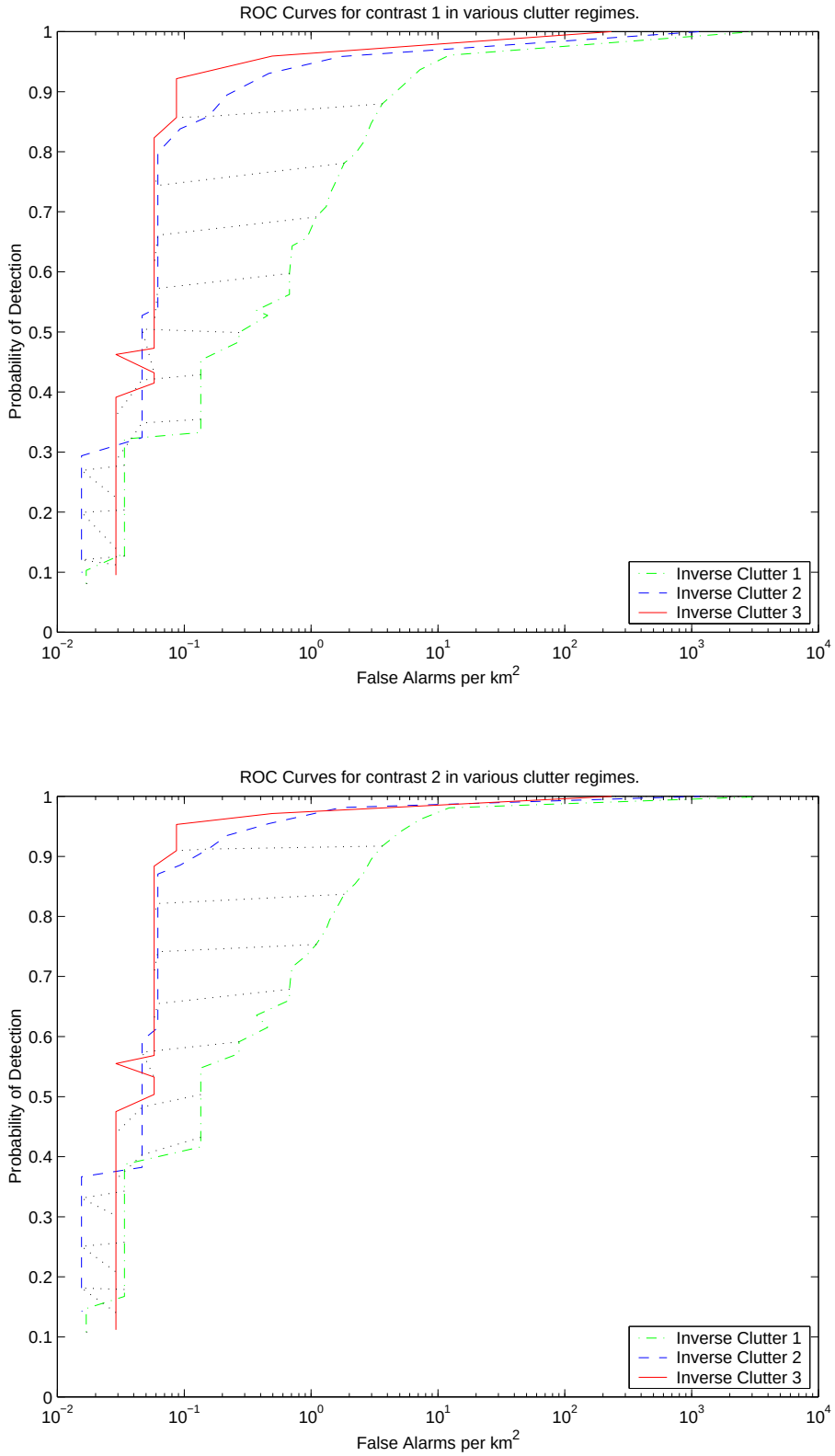


Figure 3.10: *Estimated ROC Curves for the LLC stage using 9 features*

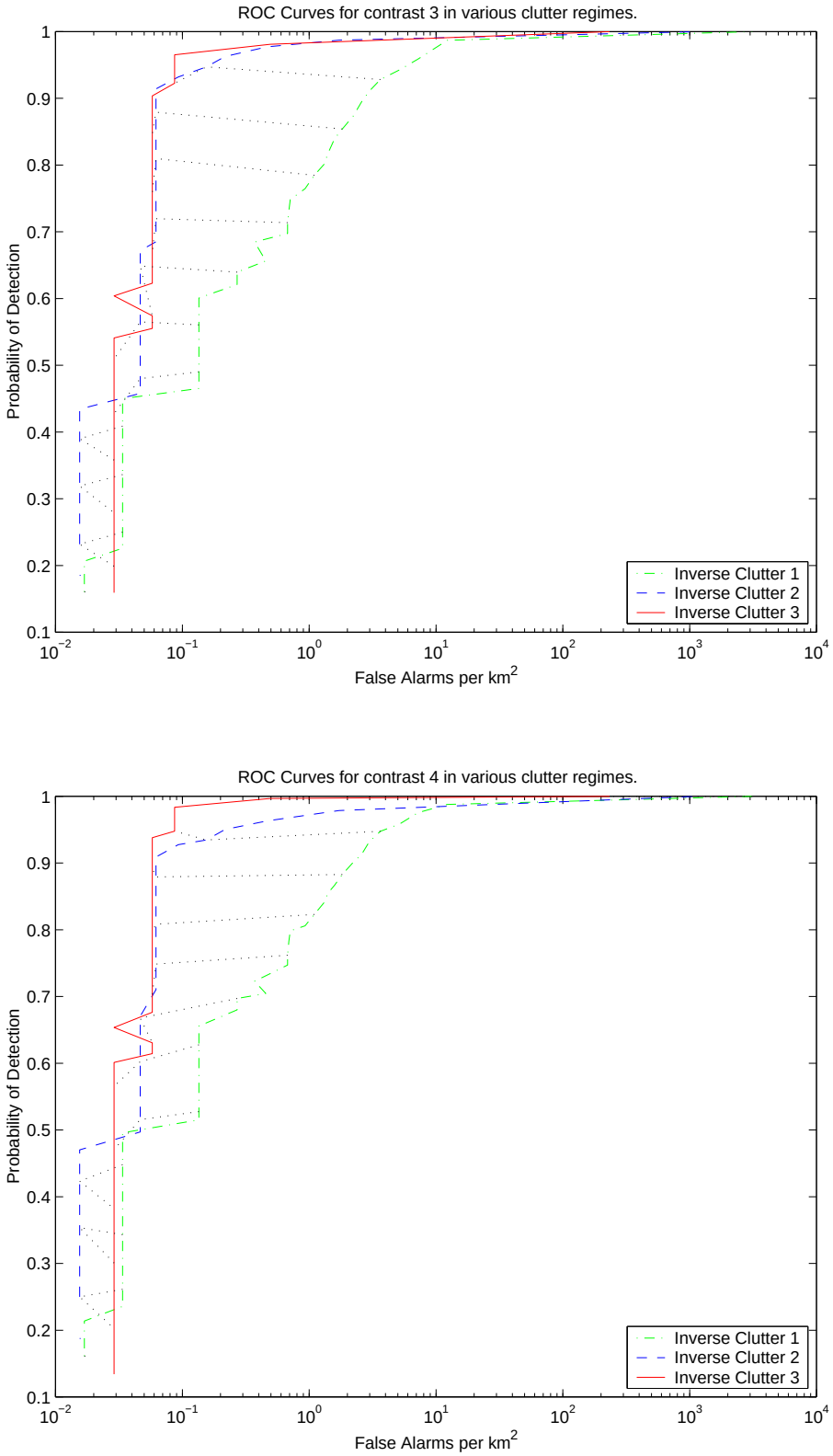


Figure 3.11: *Estimated ROC Curves for the LLC stage using 9 features*

3.6 Adaptive detection

The majority of the features discussed so far have only considered a small region centered on the target. Adaptive detection techniques compare the features of a particular area with those of its immediate background. For useful results, this technique must use an image significantly larger than the size of the target and so is quite computationally intensive. The total computational burden however could be significantly reduced if a cascaded classifier is used as suggested in the previous section. This possibility makes such a detector much more feasible.

What will be referred to as an adaptive feature, can be calculated from a normal feature by splitting a large square containing the target (say of size 64×64) into non-overlapping blocks (in this case of size 9×9). The middle of these should be centered on the suspected target. The feature to be made adaptive is then calculated over each of these blocks. If the mean and variance of the feature for the background blocks are μ_b and σ_b^2, and f is the feature of the center block, then the adaptive feature will be given by

$$\frac{(f - \mu_b)}{\sigma_b}$$

This adaptive feature can then be used in place of the original feature as a discriminant. Figure 3.12 gives an idea of the improvement in performance possible by using the adaptive features in place of their non-adaptive counterparts. Figures 3.13 and 3.14 show the performance of the adaptive versions of the 9 features used in the previous section. Once again, these results need to be verified for different data sets with non-inserted data. It does however show that a great performance improvement can be obtained by using adaptive quantities.

3.7 Conclusions and Future Directions

The results presented in this report prove that for a specific set of imagery, an LLC stage can be implemented which gives a FAR on the order of one false alarm every $25km^2$ at a PD of 90 percent for inserted targets. If this result remains true for the variety of images at different resolutions and in different whether conditions, then human operators will only need to look at small sections from 4 percent of the images. A single analyst could easily handle the required supervision, so the objectives for this LLC stage would have been met. There are however some reasons to believe that the extremely good performance seen in the test imagery may not carry over into more varied imagery.

Firstly the SVD feature, which contributes the majority of the performance to the final classifier, is not well understood. Zhang [10] has tested the features extensively, but was unable to find a physical explanation for their target detection ability. The SVD feature is related to the correlation of the image, so it is possible that it is picking up some edge information which is added during the target insertion procedure.

Another possibility is that the SVD feature is resolution dependent to some extent. At different resolutions, radar images have different correlation properties. Hence it is possible

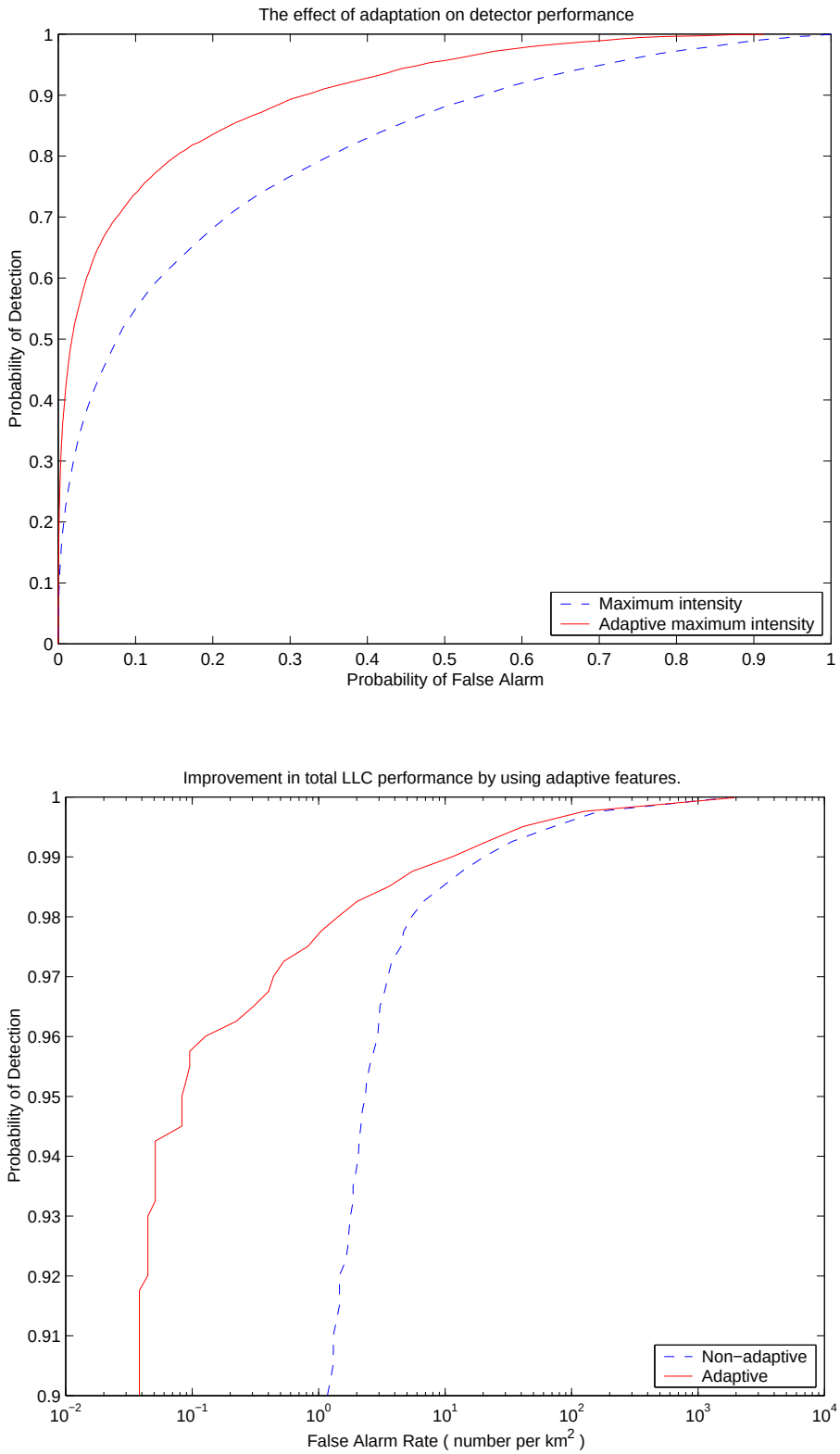


Figure 3.12: *The effect of using adaptation on the maximum intensity and the 9 best features*

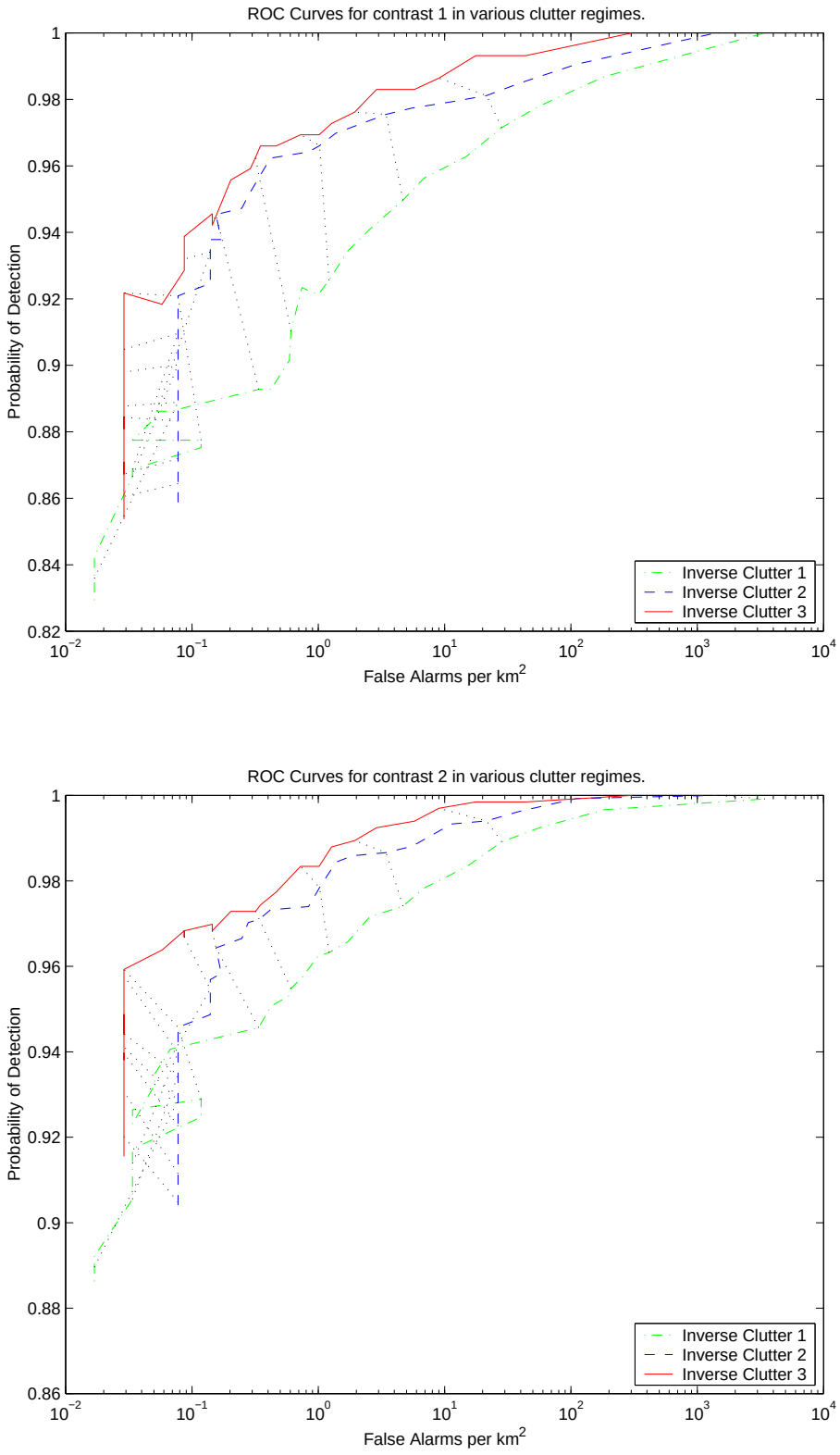


Figure 3.13: *The performance of 9 adaptive features for various clutters and contrasts*

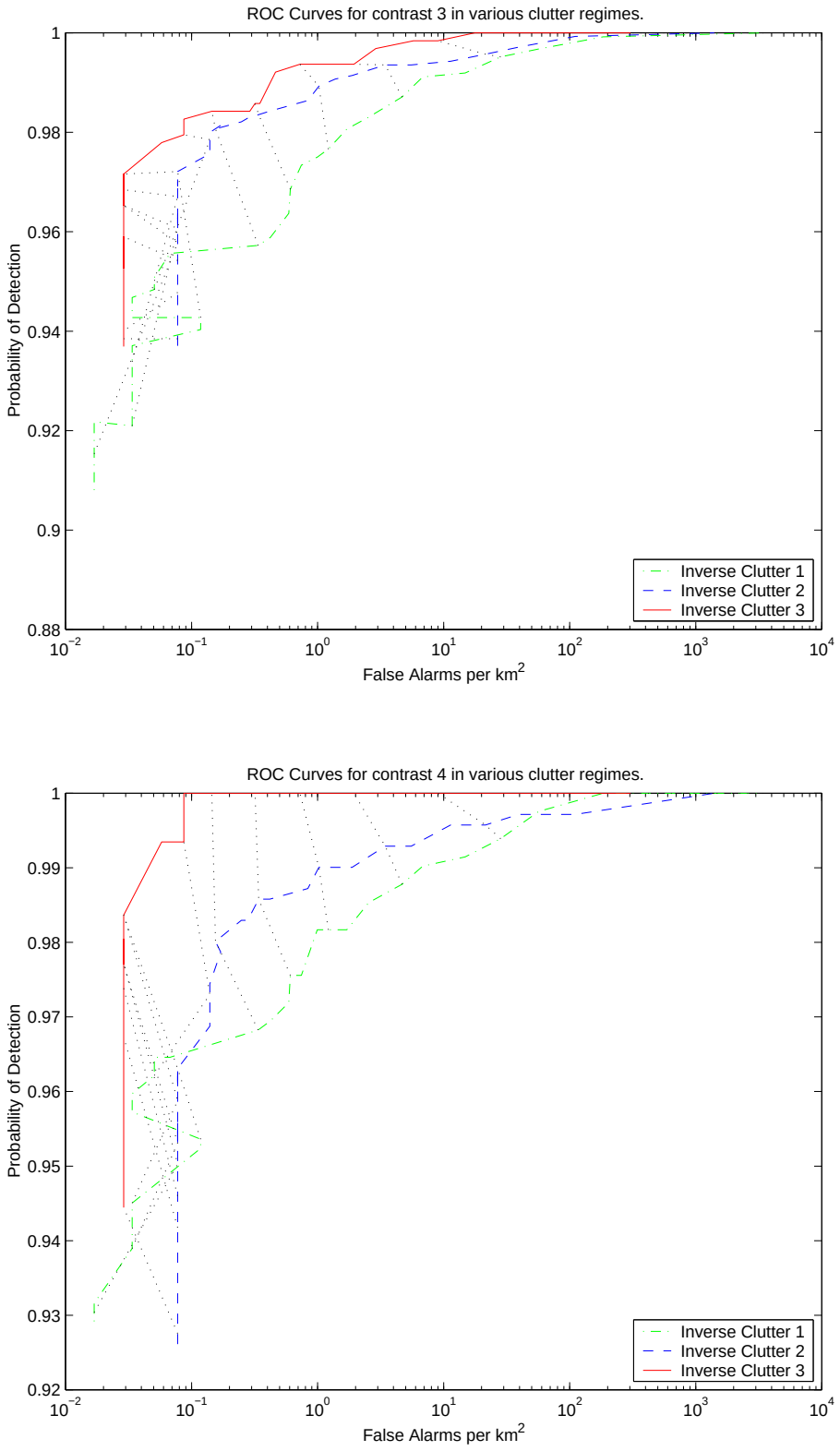


Figure 3.14: *The performance of 9 adaptive features for various clutters and contrasts*

that the SVD is detecting that the targets, which were inserted from different resolution imagery, have slightly different correlation structure. This hypothesis is given some small support by the fact that the 8th eigenvalue of an 8×8 SVD appears to have a background distribution which is completely independent of clutter and contrast (implying it is a constant of the imagery), and yet the inserted targets have a quite different distribution. Some evidence against the hypothesis is that the feature does detect four non-inserted blobs which resemble targets by eye (groundtruth is not available), but these are quite bright objects and the number of these is too small to be certain.

The second possible problem with the LLC detector is the variation in the image background due to radar configuration and weather conditions. The test data used to develop the LLC stage was limited to a single image, so a comprehensive measure of robustness could not be made. Preliminary tests indicate there can be a large variation in the FAR for different types of imagery. This variation seems to correlate to the image mean however, and tests are ongoing to see if compensating for the mean difference will produce a consistently good FAR.

While both of the above problems may ultimately reduce the system performance from those reported here, this may be partly counteracted by a third point. After examining many of the images containing inserted images by eye, it seems that many of these are so dim, or in such a cluttered background that it is unlikely that a human analyst would think they were not background. Since the ADSS project requires human supervision, it is not necessary for the LLC to detect these faint or cluttered objects, because it would probably be overruled by the image analyst. This means that the classifier threshold could be set higher and the FAR could be further reduced.

To conclude this report, it is expected that during the next six months, the LLC stage will be examined in more detail to ensure it is able to perform with unseen non-simulated data. To this end, the following tasks will be attempted:

- Test the LLC features over a variety of different imagery, and modify so that a consistent FAR can be obtained under all conditions.
- Check the LLC for a series of non-inserted targets to ensure that the classifier is not picking up insertion artifacts.
- More closely examine the feasibility of splitting the LLC stage into two cascaded detectors.
- Examine the effects of using a non-linear discriminant on the final performance of the LLC.
- Look at possible performance improvements from using features based on the complex imagery.

References

1. Redding N.J., Design of the Analysts' Detection Support System for Broad Area Aerial Surveillance, DSTO-TR-0746.

2. Cooke T.P., First Report on Features for Target/Background Classification, CSSIP-CR-9/99.

3. Cooke T.P., Discriminant based classification, CSSIP-CR-25/99.

4. Schroeder J. and Gunawardena A., "A multiplicative noise based MMSE filter for speckle reduction," Signal Processing, Special Issue on Defence Signal Processing, 1999.

5. Donohue K.D, Saniie J., Nagle D.T. and Bilgutay N.M., "Sort function analysis for order statistic filters in detection systems", SPIE Vol.1096, p.64–75, 1989.

6. Billard R.D., Arozullah M. and Barnett J.T., "Application of nonparametric detectors to point target detection in infrared clutter backgrounds", SPIE Vol.1096, p.108–118, 1989.

7. Guillemet H., Benali H., Préteux F. and Di Paola R., "Noisy fractional Brownian motion for detection of perturbations in regular textures", SPIE Vol.2823, p.40–51, 1996.

8. Kreithen D.E., Halversen S.D. and Owirka G.J., "Discriminating targets from clutter," The Lincoln Laboratory Journal, Vol.6, No.1, 1993.

9. Cooke T., Redding N., Zhang J. and Schroeder J., Target Detection Survey, CSSIP-CR-11/99.

10. Zhang J., Singular Value Features for Target Detection, CSSIP-CR-28/99.

Chapter 4

Feature extraction III

4.1 Introduction

The target detection component of JP129 contains two stages. The first is a prescreening stage which examines the entire image for targets. Due to the large amount of imagery which must be scanned the algorithm for performing this needs to be very quick, but as a result it produces a large number of false positives. To reduce the number of false alarms to a more useful level, the potential targets flagged by the prescreener are then sent to the Low Level Classification (LLC) stage.

The LLC stage involves the calculation of a number of so-called "features" for every image. Each image thus corresponds to a single point in some multidimensional feature space. An image can then be classified as target or background according to its position relative to some decision surface (the calculation of which is described in [4]) in this feature space. The current report, like the two that preceded them ([2], [3]), gives details concerning the usefulness of various features for the LLC stage.

The first two reports considered features based on magnitude only radar imagery. The performance of these features was also only tested on a single data set. This data set, while quite large, contained a small set of real targets that were artificially inserted into a wide variety of imagery, and a large number of background images that had been incorrectly classified by the prescreener. While extremely good results were reported for this set, it was also pointed out that the results were highly dependent on a single SVD feature and that this feature may have been extracting some artifact produced by the target insertion procedure. These fears turn out to have been justified, and the first section of the report presents data which confirms this hypothesis.

The remainder of the report presents feature selection and performance results for a variety of recently acquired imagery. This imagery contains real (non-inserted) targets from a variety of different radar resolutions, look angles and aspect angles. Also, some results concerning the performance benefit from the use of complex imagery and complex change detection algorithms are presented.

4.2 Testing of the SVD feature

In the previous report [3], it was found that the eighth singular value of a 9×9 block containing a potential target was extremely powerful in the detection of artificially inserted targets. These targets were real targets that were cut from other imagery and pasted with some smoothing into various clutter images, as described in Redding and Robinson [7]. As stated in the report, there was some inconclusive evidence that this SVD feature may have been detecting an artifact introduced by the target insertion process rather than the target itself. A new background data set has now been prepared to test this hypothesis.

The new background data set was obtained by constructing a 5×5 square mask, and applying a smoothing operation over the edge of the mask for each of the old background images as if there had been imagery inserted. The 9×9 SVD of the new background set was then calculated, and the distribution of the 8th singular value was compared with that of the old background set. This comparison generated the ROC curve shown in Figure 4.1 which was then plotted next to the ROC curve obtained for the inserted target data set. The ROC curves appear to be almost identical, which proves that the singular value feature was in fact mostly distinguishing smoothed images from non-smooth images rather than targets from background.

It was not only the 8th singular value which was affected by this smoothing artifact. Figure 4.2 shows ROC curves for discrimination between the various singular values in

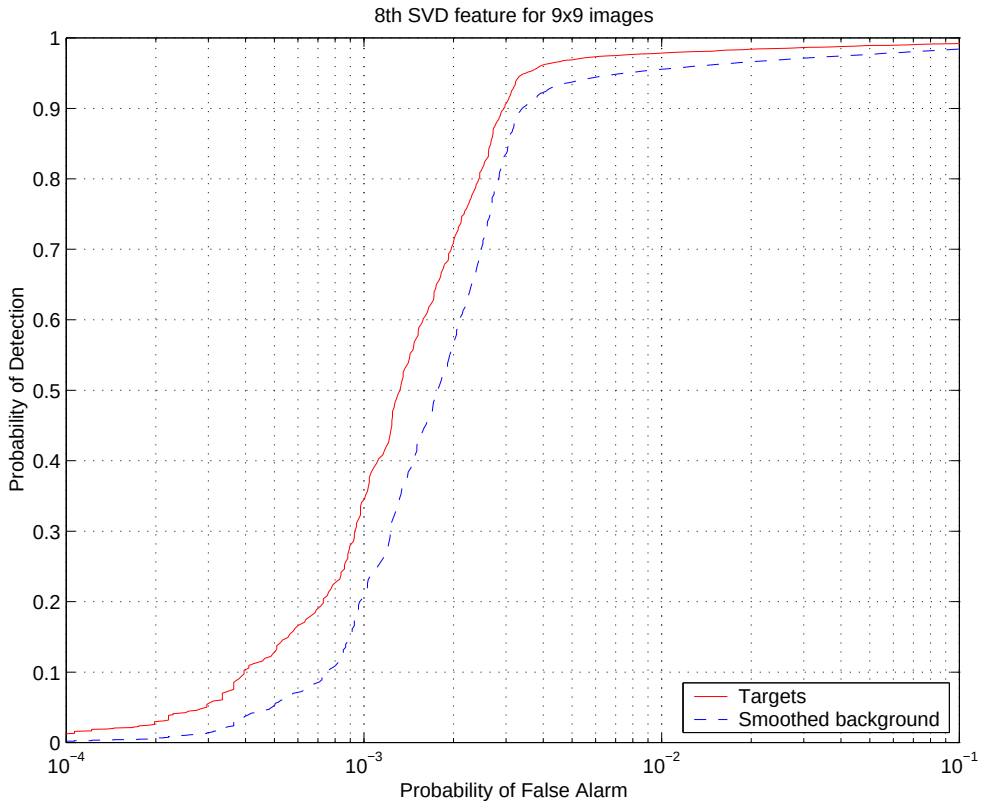


Figure 4.1: *Comparison between 8th singular value for inserted targets and smoothed background*

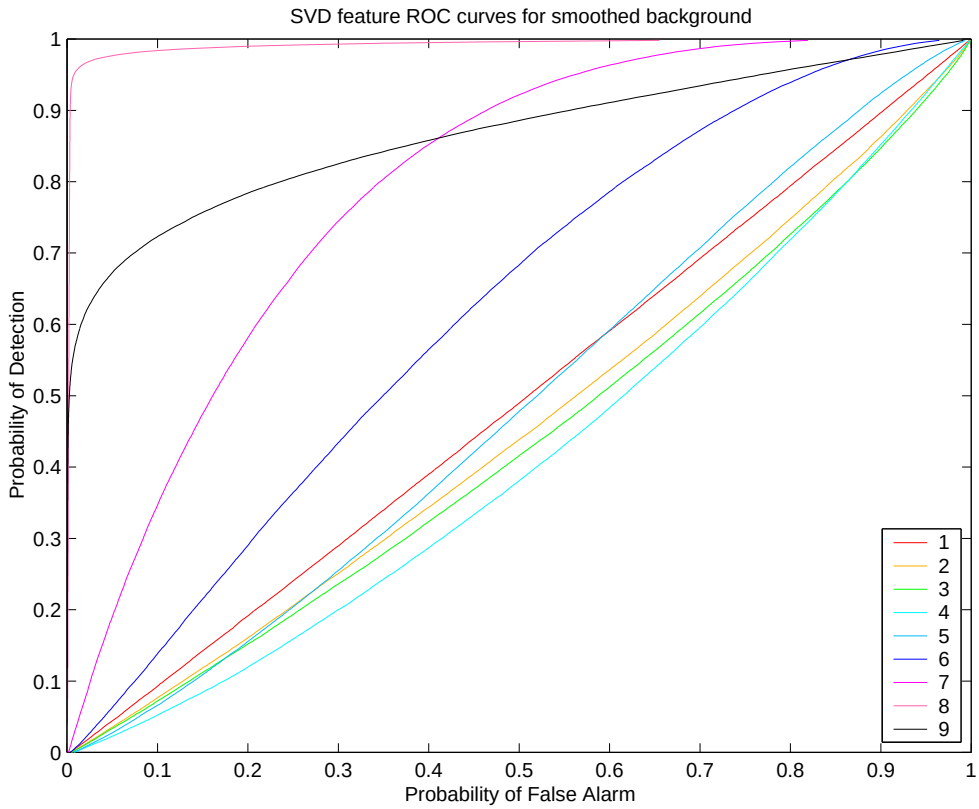


Figure 4.2: *Comparison between SVD features for inserted targets and smoothed background*

the background and smoothed background data sets. The first singular value, which is responsible for most of the image energy, is least affected by the smoothing operation so its ROC curve remains almost straight. The eighth singular value is affected the most. The exact reason for the SVD detecting this smoothing remains unclear. It is because of the possibility that inserted targets can be detected from introduced artifacts that only real data will be used in feature testing for the remainder of this study.

4.3 Magnitude data training and testing

The previous section showed one good reason why simulated data should not be used in classifier training when any other alternative is available. Considering only unsimulated data however can severely limit the amount of training data available. In this case, only data from one trial (ICT99) are available.

The ICT99 data set is divided into two parts. The first set of imagery (which shall be referred to as the spotlight imagery) was obtained by operating the INGARA radar in spotlight mode while processing the images as stripmap. The second data set was generated by operating the radar in stripmap mode to image camouflaged targets (this shall be referred to as the stripmap data).

4.3.1 Spotlight data

Imagery from the spotlight data was obtained by processing the radar returns using two different radar resolutions (1.5m and 2.0m) and for four different runs at different look angles. A prescreener [7] which had been trained on a separate data set (LS97 spotlight images which had 1m×0.7m pixels) was then applied to this imagery to extract 64×64 sized image chips. These chips were then labeled as either correctly detected targets or false alarms, and then were used in the training and testing of the low level classifier. There were 57726 backgrounds and 7006 targets in the 1.5m resolution data set, while in the 2.0m resolution data set there were 55671 backgrounds and 7277 targets.

The LLC training consisted of two stages: feature selection followed by discriminant training. Before feature selection however, it was necessary to compute the very large set of features described in the previous two reports ([2] and [3]) for each resolution. These features were calculated adaptively over 27×27 windows centered on the possible targets. The adaptive versions of each of the features were computed by subtracting the mean of that feature in the 8 outer 9×9 windows from the value of the feature in the center 9×9 window. This method of calculation was faster than that described in [3], which required a larger 63×63 window.

Once this large feature set was computed, feature selection was used to generate a more manageable smaller subset. The algorithm used was the pairwise feature selection algorithm in combination with the recursive Fisher discriminant, as described in [3]. A discriminant was then constructed for these features using a training set of half the size of the available data, and performance was tested over the rest of the data.

For the 1.5m resolution imagery, a subset of 6 adaptive features was selected. Table 4.1 shows a list describing these features. The probability of false alarm values shown are for a detection probability of 90 percent, and indicates the effect of adding individual features to the linear classifier. For instance, the first row gives the performance of the first feature alone, while the fifth row shows the false alarm rate obtained by using the first five features.

ROC curves showing the performance of the classifier obtained from these six features is shown in Figure 4.4 for each of the runs. The combined performance indicates a detection probability of about 55 percent for a false alarm rate of $1/km^2$. In contrast, repeating this process for the 2.0m data set resulted in the 11 features from Table 4.2 being selected.

From the ROC curves shown in Figure 4.4, the total performance of the LLC for the 2.0m data is a 75 percent probability of detection for a false alarm rate of $1/km^2$. This

Table 4.1: *Best 6 features for 1.5m spotlight data using* 27×27 *window*

Feature	Cumulative Pfa (percent)
Ninth row of first column of U from SVD	16.43
First row of first column of U from SVD	2.83
V parameter estimate	2.13
Skewness of the first column of V from SVD	1.39
Rank statistic 72	1.30
Fifth row of first column of V from SVD	1.20

figure is even better than the result obtained for exactly the same data at the higher 1.5m resolution. This result was quite unexpected.

One possible reason for the last result was that the feature selection method (which is by necessity sub-optimal) performed significantly better for the 2.0m data than the 1.5m

Table 4.2: *Best 11 features for 2.0m spotlight data using* 27×27 *window*

Feature	Cumulative Pfa (percent)
Fifth row of first column of U from SVD	7.63
Fifth row of first column of V from SVD	4.32
Skewness of first column of V from SVD	2.09
Rank statistic 76 - Mean	1.54
First row of first column of FFT	1.22
Rank statistic 72 - Mean	0.98
Variance of first column of V from SVD	0.77
First row of first column of V from SVD	0.62
Ninth row of first column of V from SVD	0.46
Kurtosis of first column of U from SVD	0.43
Kurtosis	0.40

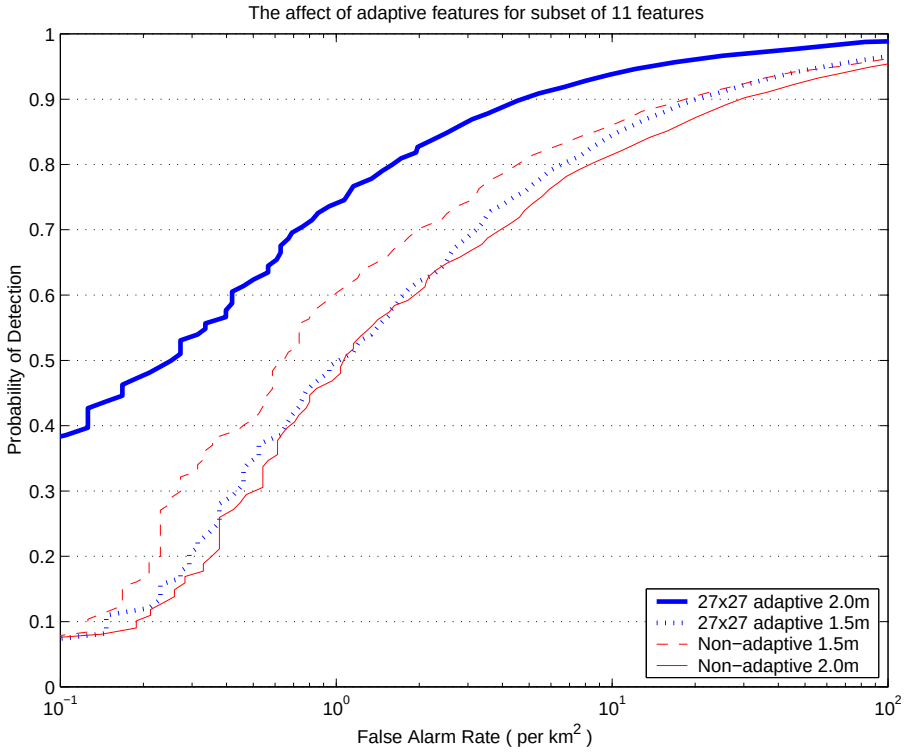


Figure 4.3: *Comparison of adaptive to non-adaptive LLC stage using 11 features for two resolutions*

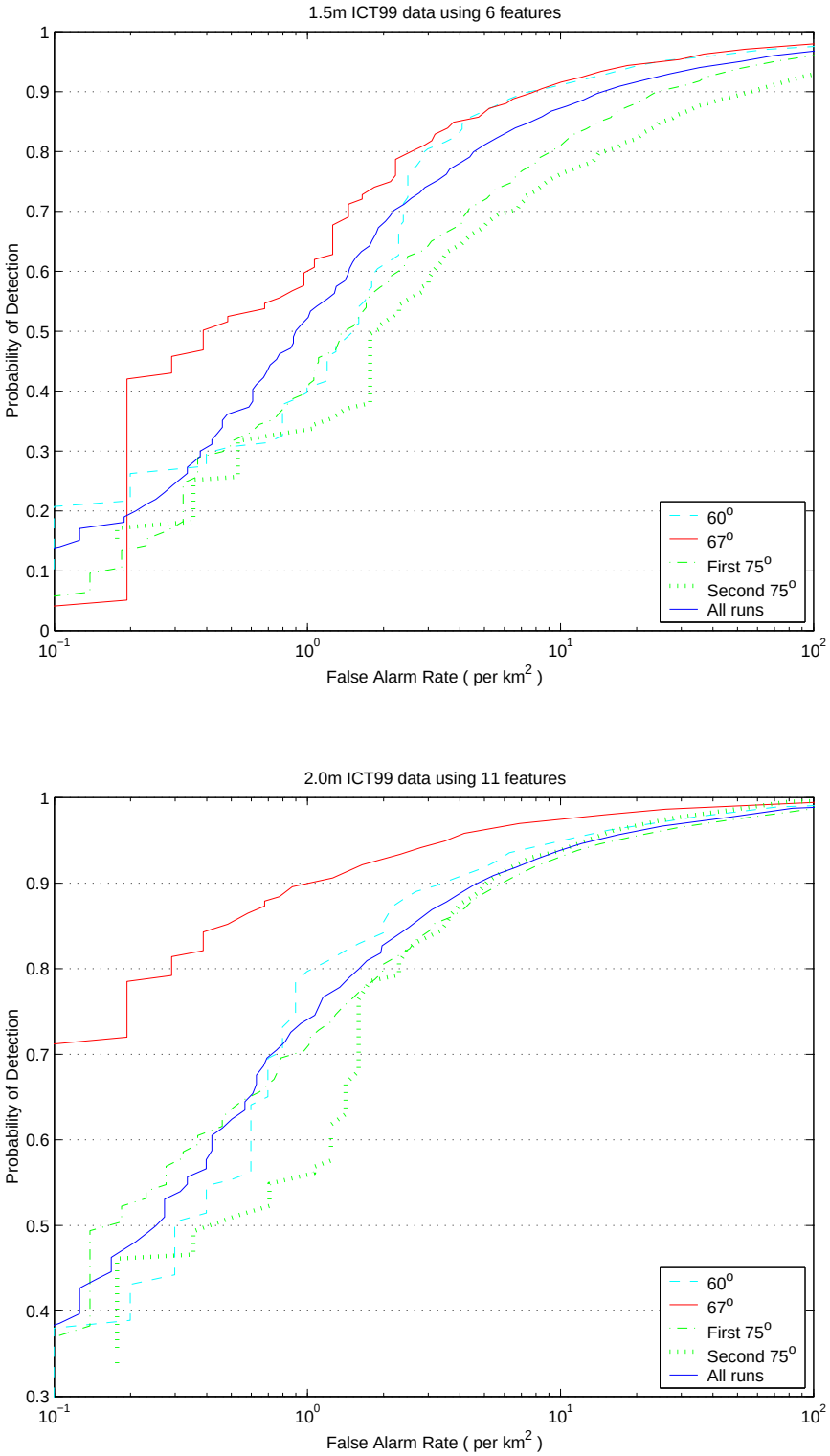


Figure 4.4: *Performance of 6 adaptive features for 1.5m spotlight data and 11 adaptive features for 2.0m data*

data. Two other possibilities were that the adaptive features were not performing as well for the 1.5m data, or that the different resolutions required different window sizes to work best. The ROC curves in Figure 4.3 compare the adaptive and non-adaptive performance of the same 11 features for 9×9 windows at the two resolutions. As expected, the non-adaptive features produced better results for the better 1.5m data. After making the features adaptive however, the 2.0m results showed a very strong improvement while the 1.5m data performance actually reduced.

The same 11 features (with some minor variations to take into account the different number of pixels per window) were used to construct the ROC curves in Figure 4.5. These show the effect of window size on the LLC performance. The performance seems to peak for a 13×13 window size. Using an adaptive detector based on this window size gave an improved detection performance as would be expected. This improved detector produced an 86 percent probability of detection for 1 false alarm per km^2 which is better than that obtained for the 2.0m data.

By performing a new feature selection for the 1.5m data with a base window size of 13×13, the performance of the new classifier was only slightly improved on that shown in Figure 4.5. The new classifier however required only 9 features to be computed instead of 11. The selected features are described in Table 4.3, with performance shown by the ROC curves in Figure 4.6.

The ROC curves in Figures 4.4 and 4.6 all show a fairly high degree of variation

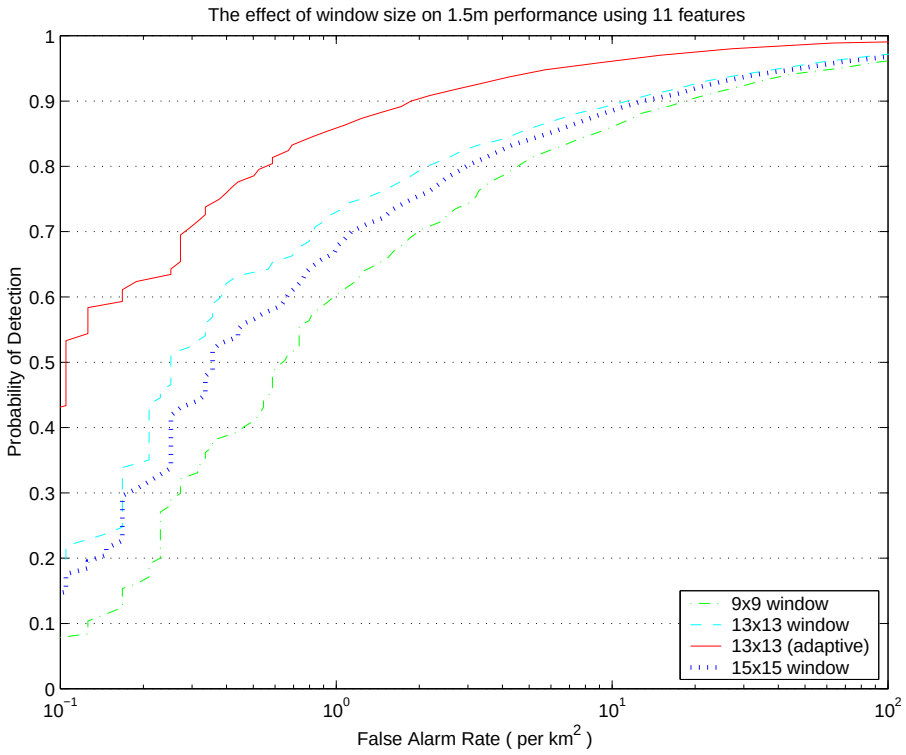


Figure 4.5: *Comparison of window sizes using 11 features at* 1.5*m resolution*

Table 4.3: *Best 9 adaptive features based on* 13×13 *window size (unless specified) for 1.5m spotlight data.*

Feature	Cumulative Pfa (percent)
Rank statistic 72 from 9×9 window	14.15
Rank statistic 100	0.66
Seventh row of first column of V from SVD	0.47
Variance of first column of V from SVD	0.28
Seventh row of first column of U from SVD	0.18
V parameter estimate from 9×9 window	0.17
13th row of first column of V from SVD	0.16
First row of first column of V from SVD	0.13
Rank statistic 157	0.12

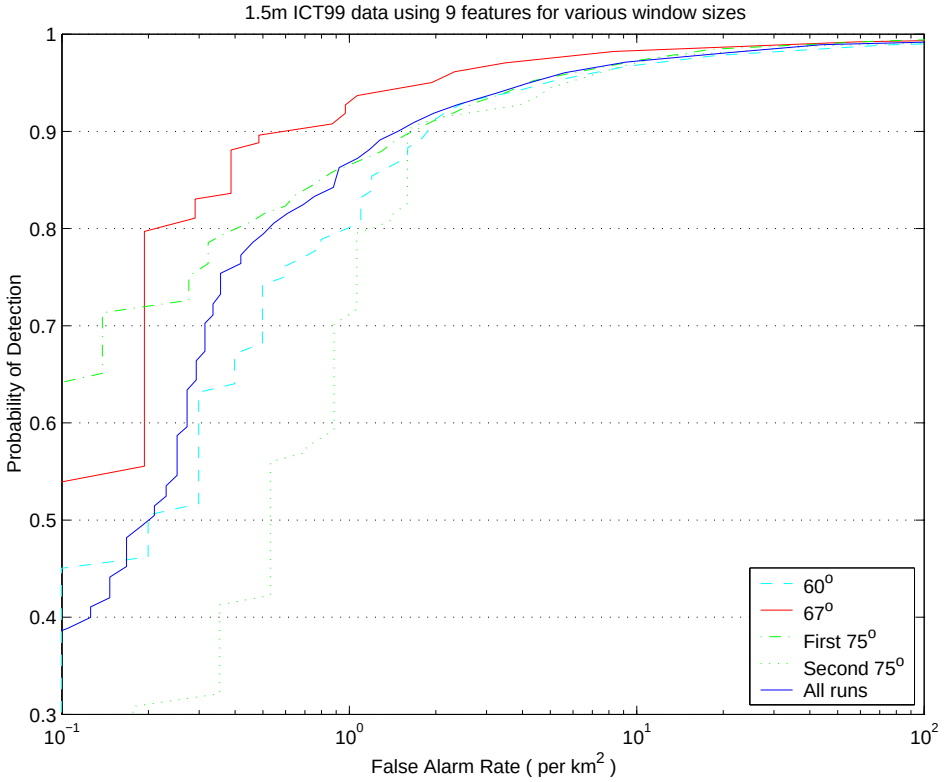


Figure 4.6: *Performance of best 9 adaptive features for 1.5m spotlight data*

between runs. Some of this variation may be due to look angle. The difference in ROC curves between the two 75 degree look angle runs indicates however that there is also some other parameter affecting the LLC performance. Due to the limited amount of available data, it is not possible to determine the source of this variation. As has probably been

mentioned by Stacy [8] [2], windy conditions can effect the amount of speckle present in imagery, so possibly small changes in wind-speed are the source of the LLC performance variation. In this case, a much larger range of imagery from these different wind speeds would be needed to ensure that the LLC results are completely robust.

4.3.2 Stripmap data

The stripmap data, was processed for three different resolutions (1.5m, 2.0m and 3.0m). Each set of imagery was compared with a previous image of the same area using a change detection algorithm [8]. Those points which showed significant differences were used to generate image chips containing possible targets. Unlike with the spotlight data, the location of the targets within the imagery had an uncertainty of up to 200m, so it was not possible to label these chips definitively as either target or clutter. Instead, each known target is associated with a small group (possibly empty) of detections from the change detection algorithm, one of which may be due to the presence of that target.

There are several consequences of not knowing the exact target position. Firstly, it is not possible to produce an exact value for the detection probability of either the change detection algorithm or the LLC stage. Estimates for the detection probability can be made by counting the number of non-empty groups of detections associated with targets divided by the number of known targets. This estimate provides a performance upper bound, which may be very inaccurate when the false alarm rate is very high. For this reason, the data set provided for LLC testing used a large change detection threshold to minimise the false alarm rate. However, it also reduced the measured prescreener detection probability to about 37 percent, limiting the test performance of the combined system. In normal operation, the threshold should be set much lower, which means that those false alarms present in the stripmap data set will be more difficult to distinguish from targets than the majority of those encountered in practice. This will also reduce the tested LLC performance.

A second point is that due to uncertainties in categorising each image chip, it is not useful to use the chips in a training set for the two-sided classifiers considered in target detection. As a result, differences in the image type (stripmap instead of spotlight modes) and weather conditions can not be taken into account to improve the classifier performance. Additionally, the targets imaged in this stripmap data set were camouflaged while no particular effort was used to conceal the targets from the spotlight imagery. All of these conditions will lead to reductions in the performance of the LLC stage on this imagery.

Keeping the above limitations into account, the ROC curves for the total system performance on the stripmap data are shown in Figure 4.7. These results were obtained by using the features selected for the spotlight data in Tables 4.3 and 4.2. The recursive Fisher discriminant was then used with both of the spotlight data resolutions to produce hyperplane decision surfaces. These surfaces were then used to estimate the detection probabilities and false alarm rates for the stripmap data. By moving the decision surface in the direction of its normal, ROC curves were generated. Figure 4.7 indicates that there is little difference between the results obtained by training of different resolution data. As

[2] Due to its restricted classification, this document has not been seen by the author

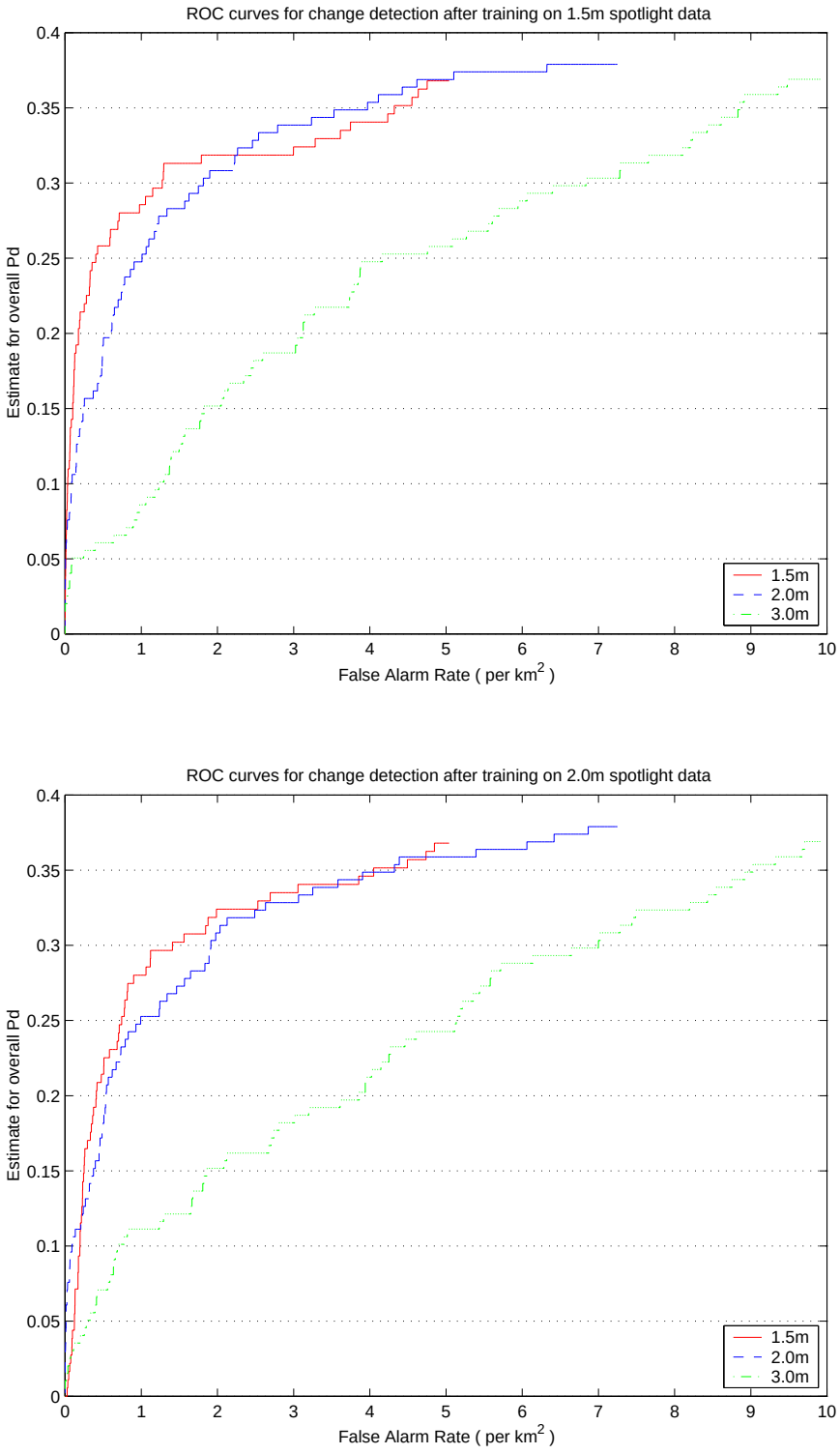


Figure 4.7: *Stripmap data ROC curves for training on 1.5m/2.0m spotlight data*

expected however, detection performance seemed to increase as the resolution of the test data set improved.

In order to generate change detection results, stripmap imagery must first be compared with a previously taken SAR image of the same area in the same orientation. After image registration, and any required rotation and interpolation, this chip from the old image which directly corresponds to the new image is referred to as the aligned image. In a report by Stacy [8] [3], it was likely to have been noted that the performance of the change detection algorithm could be improved by using more information from the aligned images.

To enhance the change detection performance, for each candidate target chip a "target mask" was calculated by thresholding the difference between the incoming image and the aligned image. It was found that when this "target mask" consisted of less than two pixels, the chip was more likely to correspond to a false alarm. Figure 4.8 shows a ROC curve for the given spotlight imagery, which was obtained by varying the threshold used to obtain the "target mask". It also shows the improvement attainable by combining this result with the features calculated for the LLC. This curve was obtained by calculating ROC curves for the LLC for various "target mask" thresholds and drawing the outer envelope of all of the curves.

Due to the testing difficulties and other limitations inherent in this data set, there are few results that can be claimed with certainty. For instance, from the figures it seems that the features do not seem to perform nearly as well as on the spotlight data. It may be however that by training the classifier on a more representative set of imagery, or by lowering the change detection threshold, that these problems may disappear. It does seem

[3]Due to its restricted classification, this document has not been seen by the author

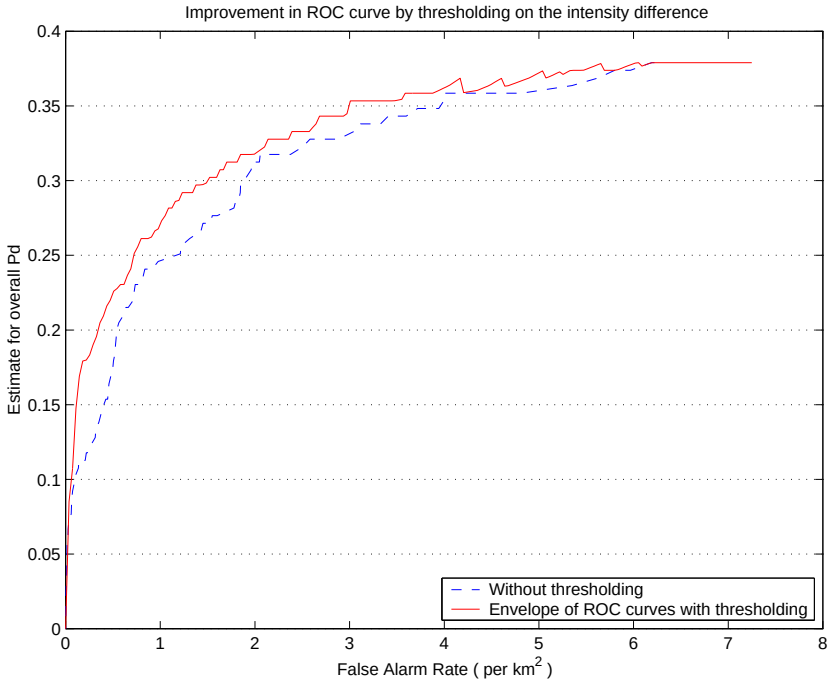


Figure 4.8: *Stripmap ROC curves obtained by changing "target mask" threshold*

though that the "target mask" threshold and the features developed for the LLC do not capture the same properties for distinguishing targets from background. As a result, when more useful data become available, it is recommended that the possibility of including this "target mask" threshold into either the prescreener or the LLC stage be investigated.

4.4 Complex Data

The radar signal return contains two components: an in-phase (I) and a quadrature (Q) component. The magnitude of the signal has an obvious physical relationship to the reflectivity of the scene being imaged, and it is this which is used to construct the SAR image. The phase however is usually completely ignored, and it was suspected that this could result in the loss of potentially useful target discriminating information.

In the first feature report [2], two classifiers using Fourier coefficient features were compared. The first used only the magnitude of the radar return, while the second used the full radar signal. A comparison showed an enormous improvement in the performance of the classifier using the complex data. This result however used only a small data set, so it was not possible to be certain the complex features were not just capturing some different aspect of the magnitude only image. To do this required a much more data.

The new data set was generated by processing the raw spotlight mode radar return to give a single look, 2.0m resolution, complex image. Then a matched filter prescreener was applied to the imagery to extract image chips, which were then labeled as either detections or false alarms. Some extra target chips which the prescreener missed were also added to the data set, and labeled as misses. In total, there were 15326 background and 794 target images in this data set.

To test the efficacy of complex features in improving target detection, it was necessary to compare the performance with that obtainable by using magnitude features only. Since single look radar imagery is of a considerably lower quality than that obtained using multiple looks, the same features that were selected previously in Table 4.2 could not be used. Hence the magnitude feature selection process was re-run for this new single-look imagery. The new features selected by this process are shown in Table 4.4.

Once the magnitude only features had been selected, a large set of complex based features was calculated for each image chip. This feature set included the magnitude of the FFT of the complex image, a complex based version of the fractional Brownian motion parameters (as described in [3]) and the multi-scale autoregressive (MAR) coefficients with

Table 4.4: *Best 5 adaptive features for single look 2.0m magnitude spotlight data*

Feature	Cumulative Pfa (percent)
Mean for 9×9 window	56.07
Energy (azimuthal) for 9×9 window	10.50
Mean for 13×13 window	3.73
(1,2) element of FFT for 9×9 window	3.35
Entropy (azimuthal) for 13×13 window	3.18

Table 4.5: *Best 7 features for single look 2.0m complex spotlight data*

Feature	Cumulative Pfa (percent)
Mean for 9×9 window	56.07
Energy (azimuthal) for 9×9 window	10.50
Mean for 13×13 window	3.73
(1,2) element of FFT for 9×9 window	3.35
α_3 from 16×16 non-adaptive MAR	3.11
α_0 from 16×16 adaptive MAR	3.12
α_0 from 16×16 non-adaptive MAR	2.77

residual errors (as described in [5]) at various window sizes. After combining this set of complex based features with the magnitude only features selected in Table 4.4, a final pairwise feature selection was used to winnow the features down to those in Table 4.5. As can be seen, at a 90 percent probability of detection, only a small improvement in the measured Pfa (from 3.18 percent to 2.77 percent) is obtained by using the complex data. Figure 4.9 shows ROC curves for both the real and complex feature sets. Both curves appear very similar, and any improvements seen in the ROC curve based on the complex data may well be due to statistical variation.

The results from Figure 4.9 indicate that there seems to be no improvement in performance from considering the complex versions of the imagery. As a double-check however, phase-difference maps were generated by comparing the phase of adjacent pixels (separate

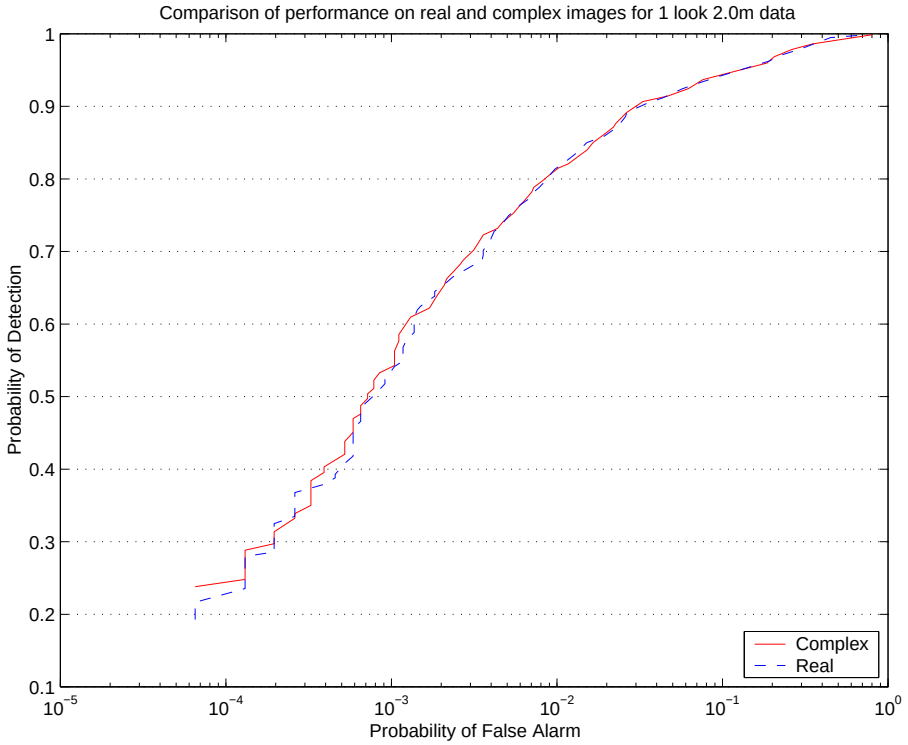


Figure 4.9: *ROC curves showing increase in performance due to consideration of complex features*

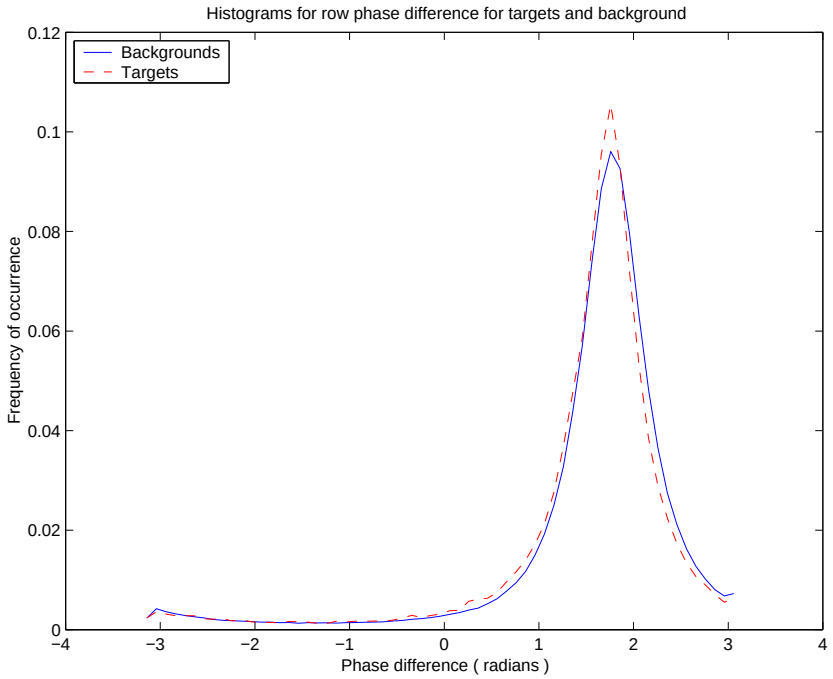


Figure 4.10: *Histograms showing azimuthal phase differences for target and background chips*

maps were created for phase differences in the range and azimuthal directions). Histograms were then generated for target and background phase differences, as are shown in Figure 4.10.

The extreme similarity in the Figure 4.10 histograms (which is also seen for phase differences in the range direction) reaffirms the hypothesis that there is little or no target information to be obtained from the complex imagery.

4.5 Direct Classification with SVMs

To distinguish between target and background images, it is theoretically unnecessary to calculate features since all of the required information is present in the pixel values of the images. Practically however, it is not possible to construct a perfect robust classifier based on the pixel values (termed direct classification) for two main reasons. Firstly there are only a finite number of samples and secondly a real classifier will have limitations on the shape of the decision surface it can generate. For these reasons, it is often better to construct feature vectors which capture the relevant image information in a form which allows easier separation. Of course, this does not imply that direct classification need be useless. It is possible that this direct classification still captures some information not present in the features, and so could be useful as a feature itself. The results in this section test this statement.

The classifier used for direct classification in this case is the Support Vector Machine (SVM) [1]. The SVM is a linear classifier which can, through the use of Mercer kernels,

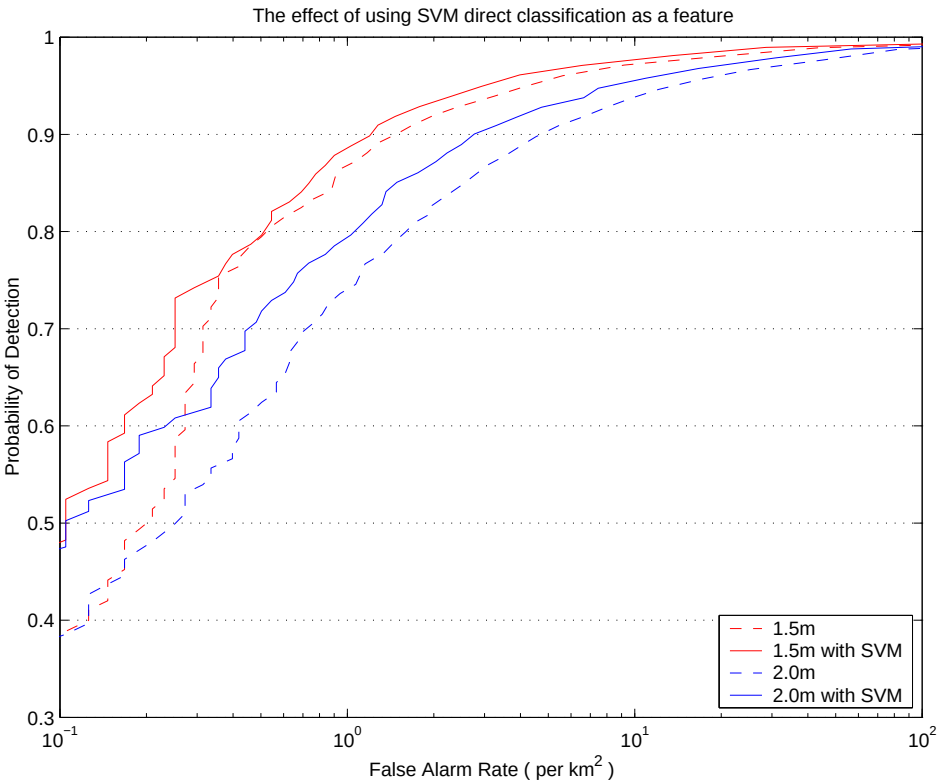


Figure 4.11: *The improvement obtained by considering direct classification using an SVM*

generate non-linear decision surfaces. By choosing different kernel functions, the SVM can generate a large variety of non-linear decision surfaces, and with correct training can give classification performances rivaling the best of other non-linear discriminants methods.

To test the usefulness of SVM direct classification in the LLC, 8×8 sized images for each resolution from the spotlight data set were used to train an SVM with a radial basis function kernel. The training and validation sets were each roughly a third the size of the complete data sets. Once the SVM had been trained to generate a decision surface, the margin of each point (which is a measure of each point's distance from the decision surface) was considered as a feature in the LLC. Figure 4.11 shows the effect of including the SVM feature along with the features selected in Section 4.3. There is a noticeable improvement in performance, implying that direct classification may have a useful role in the final LLC design.

4.6 Summary

Over the previous few reports [2] [3] and [4], a technique for creating a low level classifier has been developed. The first stage is the computation of an extensive set of features for a comprehensive training set. The first two reports ([2] and [3]) describe a large number of features of various types which could be included in this stage.

The second stage of LLC development is feature selection. There are many ways of

selecting a set of good features, but it is known that the only optimal way of doing so is a combinatoric search of all possible feature subsets. Due to the extremely large number of features considered, this is not feasible so sub-optimal techniques were considered. One of the better methods considered was the pairwise feature selection method described in [3], although MATLAB code for a faster forward-backwards selection process has also been provided. In both cases, the feature selection was accomplished by using the recursive Fisher discriminant [4] to measure the discrimination between targets and background in the feature subspaces.

The final stage for designing the LLC is the classifier. As described in [4], there are many possible ways of doing this. Due to the method of feature selection used earlier, the features seem to be chosen both for performance and to separate target and background distributions well using a linear discriminant. Hence a linear discriminant may well be the best choice for the final classifier, although a slower non-linear discriminant will usually give improved results.

Once the classifier has been designed, it is useful to have some indication of how well it will perform in practice. The current report has attempted to address that issue, although due to the limited types of imagery available it was not possible to give a definitive result. In fact, due to the high variation in performance between the imagery types seen, all of the results presented here can only be said to be accurate for the imagery on which they were tested. The performance on other imagery types will only approximate this, but the results here should at least give order of magnitude results. Given these limitations, Figure 4.12 shows ROC curves for the detection of differing contrast targets in both 1.5m and 2.0m resolution ICT99 imagery. While the performance of the contrast level 0 targets seems incredibly poor, many of these may be so dim that a human could not distinguish it from background. Due to its role as an aid for a human analyst, it is probably not necessary for the LLC to be able to detect these anyway since the human may just discard them as false alarms.

In addition to the results from Figure 4.12, the current report has also provided preliminary results on the usefulness of complex imagery. A large set of features (including complex FFT coefficients and MAR coefficients) based on the complex 2.0m single look imagery was considered in combination with a set of real features. The best classifier found for this feature set was very little better than that obtained using real features only, as was shown in Figure 4.9. This view was confirmed when single pixel displacement phase difference histograms were calculated for both target and background chips and found to be almost identical. Although this result disagreed with a result obtained in an earlier report [2], the test in the current report used a much larger data set with a wider variety of real features for comparison. As a result, none of the evidence presented in this report indicates that any worthwhile information exists in the complex imagery. While the results obtained by Zhang [10] on super-resolution seem to show some promise, it is far from clear that the method is not the equivalent of using some magnitude-only feature not previously considered.

In contrast to these negative results however, the preliminary results obtained on the change detection data set suggest that using a "target mask" threshold might improve LLC performance. Although the ROC curves in Figure 4.8 are extremely atypical of real LLC performance (due to numerous factors described in section 4.3.2), the performance

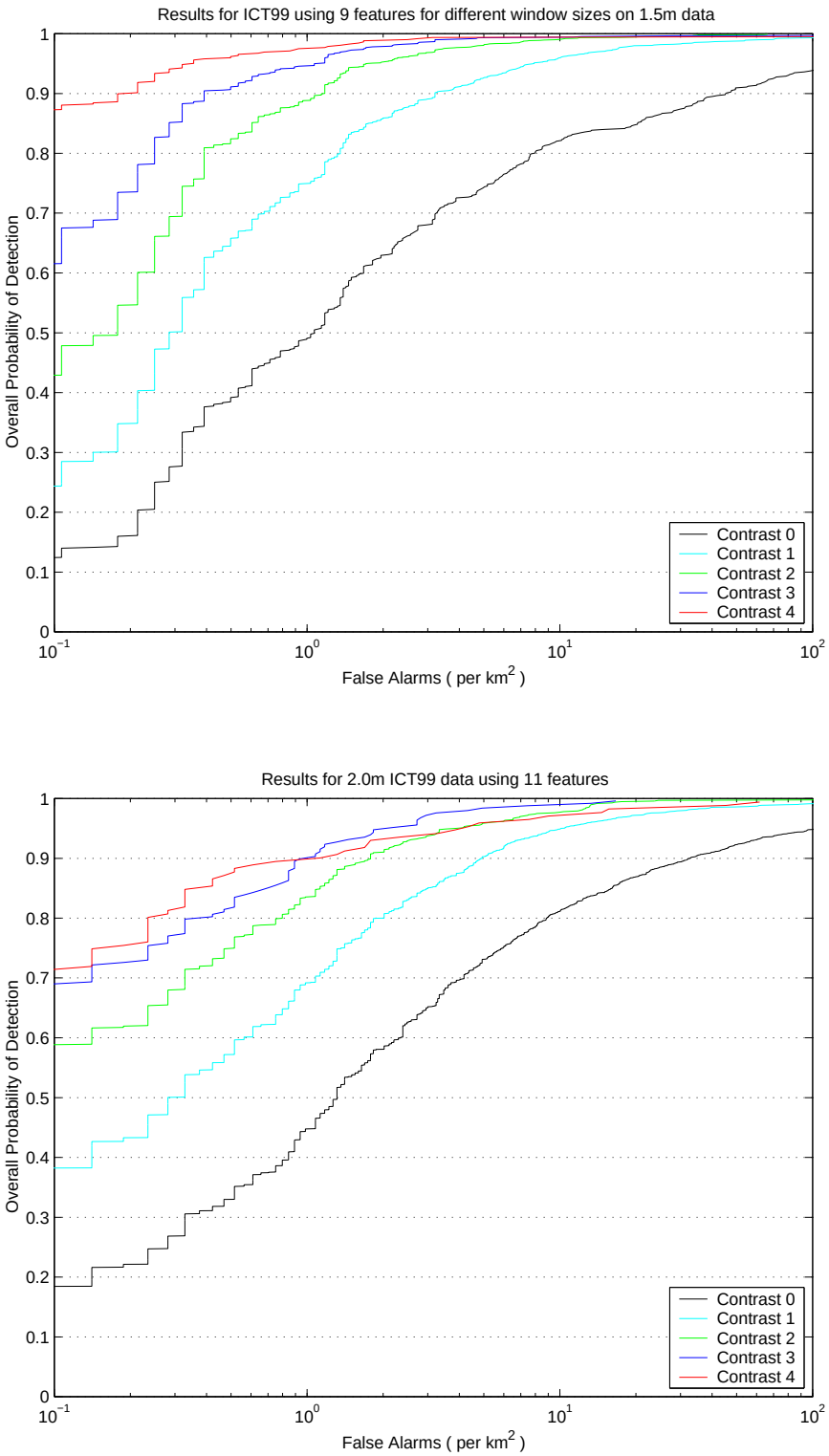


Figure 4.12: *ROC curves for different contrast targets in 1.5m and 2.0m imagery*

shows a reasonable increase when this "target mask" threshold is added to the classifier. The result however will need to be tested on a more comprehensive and better labeled data set.

4.7 Conclusion

Over the last two reports [2] [3], a method has been developed for the construction of a low level classifier (LLC). For this report, a classifier was produced and tested for a subset of the data collected in the I99 trials. Due to the limited variability of this data, the measured LLC performance shown in Figure 4.12 is not guaranteed to hold for every data set. It does however indicate that if a larger and more varied data set were available, that the false alarm rate for a 90 percent probability of detection might be about $1/km^2$ for 1.5m resolution, and $3/km^2$ for 2.0m resolution INGARA images.

In a similar project funded as part of DARPA's Warbreaker program, the Lincoln Laboratory investigation (summarised in Novak et. al. [6]) could only produce a false alarm rate of $10/km^2$ at 90 percent probability of detection for a $1m$ resolution SAR image. Now there are many differences between the Lincoln Laboratory SAR and INGARA which should be taken into account before a direct comparison can be made. For instance, the Lincoln Labs radar is fully polarimetric while INGARA is only H-H polarised. Similarly, the Lincoln Labs radar operates with a smaller stand-off range which allows a smaller duty cycle and a more accurate Doppler estimation. It also operates at a higher frequency (33GHz instead of 10GHz) so that the gain of the same sized antenna would be increased (although this may be compensated for by increased atmospheric attenuation). On the other hand, INGARA uses two looks while the Lincoln Labs imagery is likely to use only one, and the majority of the features used by the Lincoln Labs classifier were chosen specifically for their $1ft$ resolution images.

In conclusion, the classifier constructed for this data set appears to give slightly improved results to that obtained by Lincoln Labs for similar imagery. An accurate performance measurement however requires testing and retraining of the classifier over a larger and more varied data set.

References

1. Burges, C.J.C., "A tutorial on support vector machines for pattern recognition," *Data Mining and Knowledge Discovery*, **2** No.2, (1998), 1–47.

2. Cooke T., First Report on Features for Target/Background Classification, CSSIP-CR-9/99.

3. Cooke T., Second Report on Features for Target/Background Classification, CSSIP-CR-26/99.

4. Cooke T., Discriminant Based Classification, CSSIP-CR-25/99.

5. Cooke T., Redding N., Zhang J. and Schroeder J., Target Detection Survey, CSSIP-CR-11/99.

6. Novak L., Halverson S., Owirka G. and Hiett M., "Effects of polarization and resolution on SAR ATR," *IEEE Transaction on Aerospace and Electronic Systems*, **33** No.1, (1997), 102–115.

7. Redding N. and Robinson D., Prescreening Algorithm Performance in the Analysts' Detection System, DSTO-TR-0878.

8. Stacy N., Hobbs B., Nash G. and Preiss M., JP129 Indicative Capability Trial: Multimode Radar Static Target Detection, DSTO-TR-xxxx, Restricted, 2000.

9. Zhang J., Singular Value Features for Target Detection, CSSIP-CR-28/99.

10. Zhang J., Low Level Classification for Enhanced Images, CSSIP-CR-6/00.

Chapter 5

Overview of Classifiers

5.1 Introduction

The overall aim of the JP129 project is to produce a semi-autonomous system for reducing the total number of analysts required to search in real-time through the enormous amounts of SAR image data which is produced. The work of human analysts may be diminished by reducing the total amount of imagery needed to be seen by the analyst. As outlined in Redding [11], this is performed by three stages: a prescreening stage, a low level classification (LLC) stage, and a high level classification stage.

The prescreening stage [12] is applied to all of the imagery, and so needs to be very fast. Even though the majority of the background clutter is removed by this process, the remaining false alarms are still much too numerous to be of any use to an analyst. The LLC stage which follows, reduces the false alarm rate to a much more acceptable level. Since it deals only with the output from the first stage, it can be more computationally intensive than the prescreener. The number of false alarms may be reduced even further by a high level classification stage, so that an analyst need only check a small number of possible targets within a large image. Since the majority of the false alarm mitigation however is performed by the LLC stage, the overall system performance will be strongly dependent on the performance of this stage.

The LLC stage of this project will be of a similar design to that of other radar systems. For each region of interest containing a possible target, a number features are calculated. A discriminant is then used to construct a decision surface in this feature space. Regions which lie on one side of the decision surface will be classed as backgrounds and those on the other side as targets. While the choice of features probably plays the most important role in telling the difference between target and background (which is considered in Cooke [3]), the type of discriminant used to construct the decision surface is also of importance. It is the choice of this discriminant for the binary (or two class) problem which is the subject of the current report.

5.2 Linear discriminants

Linear discriminants discriminate between classes of points using a hyperplane decision surface. This type of discriminant is very popular due to their simplicity and their speed of testing. One problem with many non-linear discriminants is that they can overfit their training data, and so when they are applied to more general training data the performance degrades significantly from that on the training set. This error is referred to as the generalisation error, and can be modeled from a structural risk minimisation point of view [2] by use of the Vapnik Chervonenkis (VC) dimension. The VC dimension of a family of decision surfaces is the maximum number of points such that every labeling of those points into two classes can be separated by some particular surface from that family. For the case of linear discriminants, the VC dimension is very low, implying that they are more robust to generalisation errors than the majority of non-linear classifiers. Hence, linear discriminants can usually be applied to all of the given data, and can sometimes give improved performance over non-linear discriminants which require use of a smaller training set.

Another major advantage of linear discriminants is the lack of model parameters. Most non-linear discriminants require extra parameters to be set such as the number of Gaussians to be used for a Gaussian mixture model based discriminant, or the number of nearest neighbours to use in k-nearest neighbour discriminants. In these cases, the model parameters can only be set by trial and error, and cross-validation with a separate training and test set. Often, human supervision is required in the selection of these parameters, and the process of parameter selection may be a very long process.

To summarise, linear discriminants are generally robust, fast in both training and testing, and can be implemented completely automatically. As a result, they are ideal for testing and selecting large numbers of combinations of features, as is required for the JP129 project. The following subsections outline some commonly used and some new linear discriminants which have proved useful in the target/background classification for this project [3] [4].

5.2.1 Fisher discriminant

The Fisher discriminant [7] is probably the most commonly used linear discriminant method. The Fisher discriminant for the two class problem is derived by considering a line extending in the direction of the normal $\mathbf{n}$ to the proposed hyperplane decision surface. The projection of the classes having means μ_1, μ_2 and covariances $\mathbf{C}_1$ and $\mathbf{C}_2$ onto this line reduces the discrimination problem to a 1D problem. The two new distributions will have means $\mu_1' = \mathbf{n}^T\mu_1, \mu_2' = \mathbf{n}^T\mu_2$ and variances $\sigma_1^2 = \mathbf{n}^T\mathbf{C}_1\mathbf{n}$ and $\sigma_2^2 = \mathbf{n}^T\mathbf{C}_2\mathbf{n}$. The best discriminator should be the one which maximises the separation of these distributions in some sense. The Fisher discriminant maximises a non-parametric separation function given by

$$Separation = \frac{(\mu_2' - \mu_1')^2}{\sigma_1^2 + \sigma_2^2} = \frac{(\mathbf{n}^T(\mu_2 - \mu_1))^2}{\mathbf{n}^T(\mathbf{C}_1 + \mathbf{C}_2)\mathbf{n}}.$$

The direction of the normal to the hyperplane maximising this separation is given by

$$\mathbf{n} = (\mathbf{C}_1 + \mathbf{C}_2)^{-1}(\mu_2 - \mu_1).$$

A ROC (Receiver Operating Characteristic) curve can then be generated by considering the discrimination characteristics of the family of decision hyperplanes having the above normal. Due to the non-parametric nature of the separation functional used, the Fisher discriminant is extremely robust to distribution type. It is also incredibly fast to evaluate. In situations where the two classes have widely differing covariance structure however, the optimal linear discriminants for different parts of a ROC curve will have different normals, and in this case the Fisher can produce far from optimal results. Some slight variations on this method however can produce more acceptable results in these cases without too much extra computation.

5.2.2 Optimal linear discriminant for Gaussian classes

Anderson and Bahadur [1] derived an expression for the linear discriminant which minimises the weighted classification error for the special case when both classes are modeled by Gaussian distributions. The equation of the decision surface $\mathbf{n}.\mathbf{x} = c$ derived was given by

$$\begin{aligned} \mathbf{n} &= (\mathbf{C}_1 + \gamma\mathbf{C}_2)^{-1}(\mu_2 - \mu_1) \\ c &= \mathbf{n}^T(\mu_1 + \mathbf{C}_1\mathbf{n}) \end{aligned} \tag{1}$$

where γ is a constant whose value depends on the part of the ROC curve it is desired to operate on. This expression has a functional form very similar to that of the Fisher discriminant. In fact, the same directions of the hyperplanes can be obtained by optimising a non-parametric expression for the separation given by

$$\textit{Separation} = \frac{(\mathbf{n}^T(\mu_2 - \mu_1))^2}{\mathbf{n}^T(\mathbf{C}_1 + \gamma\mathbf{C}_2)\mathbf{n}}.$$

The extra weighting term which has been added to the expression for the separation used in the Fisher discriminant, allows a different direction for the normal of the decision hyperplane at each point. It is shown in Cooke [5] that this linear discriminant is the best possible given only the first two moments of each distribution (better discriminants are of course possible if information about the higher order moments are also available, but this will generally require a larger computation time). Due to the relative simplicity of the proof, it is presented here.

Consider a single distribution having mean μ and covariance $\mathbf{C}$. Then by projecting onto a line in the direction $\mathbf{n}$ normal to the hyperplane decision surface $\mathbf{n}.\mathbf{x} = c$, a 1D distribution is generated. This 1D distribution has mean $\mathbf{n}^T\mu$ and variance $\mathbf{n}^T\mathbf{C}\mathbf{n}$, with the decision surface becoming a simple threshold of $x = c$. Any physically meaningful error measure which depends only on the first and second order moments should be both

translation and scale invariant. From the translation invariance property, any term in the classification error involving c must only include the term $c - \mathbf{n}^T\mu$ and due to scale invariance the classification error may only be an arbitrary function of $(c-\mathbf{n}^T\mu)^2/(\mathbf{n}^T\mathbf{C}\mathbf{n})$. As a result, the weighted classification error for two class distributions will be of the form

$$f\left(\frac{(c-\mathbf{n}^T\mu_1)^2}{\mathbf{n}^T\mathbf{C}_1\mathbf{n}}, \frac{(c-\mathbf{n}^T\mu_2)^2}{\mathbf{n}^T\mathbf{C}_2\mathbf{n}}\right) \tag{2}$$

for any general function $f(x, y, \beta)$, which to be physically reasonable should be monotonically decreasing in x and y. The classification error is optimised when the derivative with respect to c and $\mathbf{n}$ are zero. Differentiating with respect to c gives

$$\begin{aligned} & f_x\left(\frac{(c-\mathbf{n}^T\mu_1)^2}{\mathbf{n}^T\mathbf{C}_1\mathbf{n}}, \frac{(c-\mathbf{n}^T\mu_2)^2}{\mathbf{n}^T\mathbf{C}_2\mathbf{n}}\right)\frac{2(c-\mathbf{n}^T\mu_1)}{\mathbf{n}^T\mathbf{C}_1\mathbf{n}} \\ + \quad & f_y\left(\frac{(c-\mathbf{n}^T\mu_1)^2}{\mathbf{n}^T\mathbf{C}_1\mathbf{n}}, \frac{(c-\mathbf{n}^T\mu_2)^2}{\mathbf{n}^T\mathbf{C}_2\mathbf{n}}\right)\frac{2(c-\mathbf{n}^T\mu_2)}{\mathbf{n}^T\mathbf{C}_2\mathbf{n}} = 0, \end{aligned}$$

which after simplification yields

$$\frac{f_x(..\,,..)}{f_y(..\,,..)} = -\frac{\mathbf{n}^T\mathbf{C}_1\mathbf{n}}{\mathbf{n}^T\mathbf{C}_2\mathbf{n}}\,\frac{c-\mathbf{n}^T\mu_2}{c-\mathbf{n}^T\mu_1}. \tag{3}$$

Differentiating (2) with respect to $\mathbf{n}$ produces

$$\begin{aligned} & f_x\left(\frac{(c-\mathbf{n}^T\mu_1)^2}{\mathbf{n}^T\mathbf{C}_1\mathbf{n}}, \frac{(c-\mathbf{n}^T\mu_2)^2}{\mathbf{n}^T\mathbf{C}_2\mathbf{n}}\right)\left(\frac{(c-\mathbf{n}^T\mu_1)\mu_1}{\mathbf{n}^T\mathbf{C}_1\mathbf{n}} + \frac{(c-\mathbf{n}^T\mu_1)^2\mathbf{C}_1\mathbf{n}}{(\mathbf{n}^T\mathbf{C}_1\mathbf{n})^2}\right) \\ + f_y & \left(\frac{(c-\mathbf{n}^T\mu_1)^2}{\mathbf{n}^T\mathbf{C}_1\mathbf{n}}, \frac{(c-\mathbf{n}^T\mu_2)^2}{\mathbf{n}^T\mathbf{C}_2\mathbf{n}}\right)\left(\frac{(c-\mathbf{n}^T\mu_2)\mu_2}{\mathbf{n}^T\mathbf{C}_2\mathbf{n}} + \frac{(c-\mathbf{n}^T\mu_2)^2\mathbf{C}_2\mathbf{n}}{(\mathbf{n}^T\mathbf{C}_2\mathbf{n})^2}\right) = 0. \end{aligned}$$

Dividing through by $f_y(..\,,..)$ and then substituting equation (2) makes

$$\begin{aligned} - \quad & \frac{\mathbf{n}^T\mathbf{C}_1\mathbf{n}}{\mathbf{n}^T\mathbf{C}_2\mathbf{n}}\left(\frac{c-\mathbf{n}^T\mu_2}{c-\mathbf{n}^T\mu_1}\right)\left(\frac{(c-\mathbf{n}^T\mu_1)\mu_1}{\mathbf{n}^T\mathbf{C}_1\mathbf{n}} + \frac{(c-\mathbf{n}^T\mu_1)^2\mathbf{C}_1\mathbf{n}}{(\mathbf{n}^T\mathbf{C}_1\mathbf{n})^2}\right) \\ + \quad & \left(\frac{(c-\mathbf{n}^T\mu_2)\mu_2}{\mathbf{n}^T\mathbf{C}_2\mathbf{n}} + \frac{(c-\mathbf{n}^T\mu_2)^2\mathbf{C}_2\mathbf{n}}{(\mathbf{n}^T\mathbf{C}_2\mathbf{n})^2}\right) = 0. \end{aligned}$$

After some elementary algebra, this gives

$$\left[\frac{c-\mathbf{n}^T\mu_1}{\mathbf{n}^T\mathbf{C}_1\mathbf{n}}\mathbf{C}_1 - \frac{c-\mathbf{n}^T\mu_2}{\mathbf{n}^T\mathbf{C}_2\mathbf{n}}\mathbf{C}_2\right]\mathbf{n} = \mu_2 - \mu_1 \tag{4}$$

which after some manipulation can be seen to be equivalent to equations (1), where $\gamma = [(\mathbf{n}^T\mathbf{C}_1\mathbf{n})(c - \mathbf{n}^T\mu_2)]/[(\mathbf{n}^T\mathbf{C}_2\mathbf{n})(c - \mathbf{n}^T\mu_1)]$. The exact value for γ can be obtained from equation (3) which determines the ROC operating point for the discriminant. Hence, the family of hyperplanes (1) obtained by varying γ will optimise the classification error based on the first and second moments of each distribution, regardless of its exact functional form. This will result in the optimal linear discriminant for any sets of class probability distributions which vary only with Mahalanobis distance (such as Gaussian or students-t distributed classes). It will also minimise the maximum possible classification error for a given set of first and second order moments.

5.2.3 1D parameter search

The only sure method for obtaining the best possible linear discriminant for a given set of classes with unknown distribution is an exhaustive search through all possible linear discriminants. While this may be feasible for low dimensional data, the computational difficulty increases exponentially with the number of features. For large numbers of features, this quickly becomes infeasible, and so faster sub-optimal techniques must be used. One possible time-saver is to decrease the dimensionality of the search required by examining sets of 'most likely' directions for the normal of the decision surface.

As seen in the previous section, the set of hyperplanes $\mathbf{n}.\mathbf{x} = c$ defined by equations (1), were the best guesses possible based only on the first and second moments of each distribution. If instead of using a fixed c for a particular direction $\mathbf{n}(\gamma)$, a 1D search was used to determine the optimal value of c, then the resulting discriminant could not be any worse than the original discriminant, and for skewed distributions may give a significantly improved performance without the need for a great deal more computation. Since when $\gamma = 1$, this is the special case of the Fisher discriminant, then this method will have a lower bound equal to that of the Fisher discriminant.

5.2.4 Recursive Fisher

The previously described methods rely only on the global shape information of each of the classes being discriminated. Although this will usually provide a good rough estimate for the position of the optimal decision hyperplane, it is the local shape of each class which is the biggest determining factor in its exact placement. Several techniques such as Support Vector Machines (SVMs) and boosting also make use of this local information, but they do not exploit it fully. Both of these methods apply weighting factors to all misclassified points from an initial rough estimate, not just those close to the decision surface. The recursive Fisher method which is presented here utilises the idea of support vectors, and retains all of the advantages of the original Fisher method (high speed and robustness). It also appears to perform extremely well, consistently produce better performing discriminants than an SVM with a linear kernel, despite the extra computational complexity of the SVM. It also has the ability to be extended to non-linear discriminants through the use of Mercer kernels, as explained later in the section on non-linear discriminants.

The algorithm for the linear version of the Recursive fisher method can be defined as follows:

1 *Initialisation:* Set the percentage of support vectors S to 90 and calculate the initial decision surface $\mathbf{n}.\mathbf{x} = c$ by using the Fisher discriminant to calculate $\mathbf{n}$ and choosing c to satisfy your required optimality condition (for instance, a fixed classification error).

2 *Choosing support vectors:* Generate two new distributions by keeping the closest S percent of points from each class to the decision surface $\mathbf{n}.\mathbf{x} = c$.

3 *Fisher discriminant:* Calculate the new decision surface by finding the Fisher discriminant of these two new distributions to determine $\mathbf{n}$ and again choose c to satisfy the optimality condition.

4 *Loop termination condition:* Decrease the percentage of support vectors S by 10 percent. If S is zero, then end the loop, otherwise go to step 2.

In this algorithm, the percentage decrement in the number of support vectors was set to 10 percent somewhat arbitrarily. Smaller decrements could be used which should improve performance, but this would also increase the number of iterations required and thus the computation time.

For univariate distributions, the best discriminant is usually obtained for the lowest percentage of support vectors. For the multivariate case however, decreasing the percentage of support vectors can in fact significantly degrade discrimination. To prevent this, the performance on the training data should be measured each time through the loop and the decision surface which gives the best results should be used. In this way, since the initial discriminant is the Fisher discriminant, this technique will always give a smaller error on the training data than the Fisher discriminant.

5.2.5 Support Vector Machines (linear version)

A Support Vector Machine (SVM) [2] is a method for calculating linear discriminants which uses only information about certain points from the point distribution, which are referred to as support vectors. It is a fairly computationally expensive technique, having a computational complexity of $\mathcal{O}(N^3)$ for N training points. It does however use all of the information about the known points instead of just the first two moments as for the Fisher discriminant. It is also guaranteed to converge although the decision surface to which it converges may not be optimal in the sense of having the best false alarm rate (FAR) for a given PD. The biggest advantage of this technique is its extension to non-linear discriminants through the use of Mercer kernels, although similar techniques can be applied to other linear discriminants such as the Kernel Fisher Discriminant [9]. These kernel techniques will be discussed later in the section on non-linear discriminants. For the time being, only the linear theory shall be explained.

Suppose the set of training points $\mathbf{x}_i$ is associated with a class y_i which is -1 if the point belongs to the first class, and $+1$ otherwise. Then consider a function $f(\mathbf{x}) = \mathbf{n}.\mathbf{x}+b$ which categorises points $\mathbf{x}$ as being of the first class where $f(\mathbf{x}) < -1$ and of the second class where $f(\mathbf{x}) > 1$. Points satisfying $-1 < f(\mathbf{x}) < 1$ are of indeterminate class (although for the final classifier, the hyperplane $f(\mathbf{x}) = 0$ is the decision surface). The hyperplanes

along which equality occurs are referred to as the margins. In the case of separable classes, an SVM will maximise the distance between the distributions (*i.e.* maximising the size of the distance between the margins, or minimising the reciprocal of the distance), subject to the constraint that the classes are classified correctly.

The previous paragraph can be expressed in mathematical notation as

$$\begin{aligned} \text{Minimise:} \quad & \frac{1}{2}||\mathbf{n}||^2 \\ \text{Subject to:} \quad & y_i f(\mathbf{x}_i) \geq 1 \ \text{ for all } i. \end{aligned}$$

In general however, the classes will not be separable, and this can be taken into account through the use of slack variables. By shifting each misclassified point $\mathbf{x}_i$ by a distance ϵ_i, the classes can be forced to become separable. To minimise the total amount of shifting that needs to occur, an extra term is added to the minimisation problem. A regularisation parameter C is also included in this term, to allow different trade-offs between the distribution separation and misclassification. The new problem becomes

$$\begin{aligned} \text{Minimise:} \quad & \frac{1}{2}||\mathbf{n}||^2 + C\sum_{i=1}^{N}\epsilon_i \\ \text{Subject to:} \quad & y_i f(\mathbf{x}_i) \geq 1 - \epsilon_i \ \text{ for all } i \\ & \epsilon_i \geq 0 \ \text{ for all } i. \end{aligned}$$

This is a convex quadratic optimisation problem, which means it has a unique optimal solution for $\mathbf{n}$. The solution will also only depend on those points having non-zero slip variables (*i.e.* those points that are misclassified), and these are termed support vectors. By using Lagrange multipliers, it can be shown that the above problem is equivalent to solving the dual problem.

$$\begin{aligned} \text{Maximise:} \quad & \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{N}\alpha_i y_i \alpha_j y_j \mathbf{x}_i.\mathbf{x}_j \\ \text{Subject to:} \quad & \sum_{i=1}^{N}\alpha_i y_i = 0, \ 0 \leq \alpha_i \leq C. \end{aligned}$$

and the decision function becomes (after using Karush-Kuhn-Tucker conditions to evaluate the bias b),

$$f(\mathbf{x}) = \sum_{i=1}^{N}\alpha_i y_i \mathbf{x}_i.\mathbf{x} + b.$$

This problem can now be solved using quadratic programming to yield solution optimising the separation functional. It does not however necessarily give the best FAR

for a given PD, which would be much more useful. It also requires determination of the best regularisation parameter C, which as yet can only be done by trial and error using numerous runs and a validation set as well as a training set.

Another problem with this method, is that it gives a single decision surface corresponding to one possible operating point of a detector. This can be fixed by splitting the regularisation parameter C into separate constants for each class C_1 and C_2. By varying the ratio of these constants, different operating points can be achieved. It does however make choosing the best sets of parameters more time consuming.

While SVM linear discriminants generally give a better FAR for a given PD than many other linear discriminants, the numerical results given later seem to indicate that it performs roughly as well as the recursive Fisher discriminant described earlier, but is a lot slower and requires supervised training and the use of validation sets. It does however become more competitive with other discriminants when used in its non-linear discrimination mode as will be commented on later.

5.3 Non-linear discriminants

For two classes having known probability distributions $p_1(\mathbf{x})$ and $p_2(\mathbf{x})$ in feature space, the optimal detector will be the Neyman-Pearson detector. This classifies a point as belonging to class 1 if the likelihood ratio $p_1(\mathbf{x})/p_2(\mathbf{x})$ is larger than some threshold λ, which may be varied to generate the optimal ROC curve. In general, the class probability distributions are not known, and the optimal family of decision surfaces can only be estimated. While linear discriminants are generally fast to compute and robust, they can only crudely model the actual optimal solution. Non-linear discriminants however are much more versatile, but generally also more computationally intensive. Also, due to their higher VC dimension, they have a tendency to overfit their training data. With careful supervised training however, non-linear discriminants have the potential to produce the best classification results.

5.3.1 Quadratic and Gaussian Mixture Model Discriminants

The Gaussian or normal distribution is probably the most widely studied of all distributions, and models a wide variety of processes.

Given a set of N samples $\mathbf{x}_i$ from a multidimensional Gaussian, the best possible estimate for its distribution function will be

$$p(\mathbf{x}) = \frac{1}{\sqrt{2\pi|\mathbf{C}|}} \exp\left(-\frac{1}{2}(\mathbf{x}-\mu)^T\mathbf{C}^{-1}(\mathbf{x}-\mu)\right)$$

where the parameter estimates for the mean and covariance are

$$\mu = \frac{1}{N}\sum_{i=1}^{N}\mathbf{x}_i$$

$$\mathbf{C} = \frac{1}{N-1}\sum_{i=1}^{N}(\mathbf{x}_i - \mu)(\mathbf{x}_i - \mu)^T.$$

By modeling both of the classes to be discriminated with Gaussians, an estimate of the likelihood ratio can be calculated and thresholded to produce a decision surface. This type of classifier is known as a quadratic discriminant, since the decision surface is a quadratic function of $\mathbf{x}$. This classifier is computationally extremely quick, and will be the optimum if the features used are in fact Gaussian. Deviations from this model however can easily result in much worse discrimination than any of the linear discriminants presented in the last section.

Another way to model the probability distributions of each of the classes is to express the probability density functions as

$$f(\mathbf{x}) \approx \sum_{i=1}^{N} \alpha_i \exp\left(-\frac{1}{2}(\mathbf{x}-\mu_i)^T\mathbf{C}_i^{-1}(\mathbf{x}-\mu_i)\right).$$

Expressing each distribution as a weighted sum of Gaussians is known as a Gaussian mixture model, and while it is mostly useful for modeling multi-modal distributions, some improvement can also be gained for non-Gaussian distributions having single peaks. When using N Gaussians to fit d dimensional probability distributions, the unknown quantities α_i, μ_i and $\mathbf{C}_i$ constitute $N(1+d+d^2)$ individual variables to be determined from a training set. Estimates of these quantities can be found by maximising the probability that the training set is actually produced by the Gaussian mixture. This will involve non-linear optimisation, and cannot be solved exactly, so numerical methods such as steepest ascent must be employed to give a suboptimal solution, as described in Jarrad and McMichael [8]. Once the Gaussian mixture model has been determined, then a point may be classified as belonging to one of the two classes by calculation of the estimated likelihood ratio obtained from the model, and then comparing to a threshold.

As with all non-linear discriminants, this Gaussian mixture based classifier is fairly computationally intensive compared with most linear discriminants. It is also somewhat distribution dependent, since distributions with tails that differ in shape from that of a Gaussian may not be modeled well using a Gaussian mixture. This can be circumvented slightly, since the method outlined in [8] can be extended to a more general set of basis functions which are monotonically decreasing with Mahalanobis distance (which for a distribution having mean μ and covariance $\mathbf{C}$ is defined as $(\mathbf{x}-\mu)^T\mathbf{C}(\mathbf{x}-\mu)$ for a point x) such as a student-t distribution. There still remains a large class of distributions which remain difficult to model using this method.

Another difficulty with the Gaussian mixture discriminant is the choice of number of mixtures with which to model the classes. The solution to this problem will be dependent on the problem to be solved, and can only be satisfactorily resolved by finding the test error after testing on a separate training set.

5.3.2 K-Nearest Neighbour Discriminants and Vector Quantisation

The K-nearest neighbour classifier is probably the most used of all non-linear discriminants. As the name suggests, this discriminant classifies a point $\mathbf{x}$ in feature space by computing the distances to points within a given training set. The K points from this set which are closest in some sense (by Euclidean, weighted Euclidean, Mahalanobis or any of a number of other metrics) to the point $\mathbf{x}$ are then used to classify the point. For K odd, this is often done by choosing the class to which over half of the K belong as the classification of $\mathbf{x}$. One nice feature about this algorithm is that as the number of training points becomes very large, it can be proven that the misclassification rate r is bounded by twice the minimum possible (or Baye's) misclassification rate ρ [6]. In fact

$$r \leq \rho(2 - \rho).$$

In its simplest form, this algorithm has a few drawbacks. While only one parameter is required to be determined, which is a lot less than for most non-linear discriminants, only one possible decision surface is possible. This means that only a single operating point on a ROC curve can be found. One solution to this problem could be to weight the distances from the points of each class differently.

Another problem is the inherent assumption of isotropy and homogeneity of the classes in the feature space. Because a straight Euclidean distance is being used for the calculation of the nearest neighbours, a more important feature is not weighted with any more importance than a less important one. This can be taken into account somewhat by appropriate prescaling of the classes in the feature space, but it does not take into account the possibility that the most important feature varies depending on the position in the feature space. This limitation of the algorithm is a feature of many non-linear methods such as neural networks and support vector machines with radial basis function kernels.

One of the most significant aspects of this method is probably the testing time. Once a discriminant has been trained, to determine the class of an unknown vector, the distance of the vector to every point in the training set must be calculated. For large numbers of points this can be quite computationally time-consuming. This time may be reduced by using a technique known as Vector Quantisation (VQ) which models the classes by N points $\mathbf{q}_i$ in such a way as to minimise the distortion which is given by

$$\mathit{Distortion} = \frac{1}{N} \sum_{i=1}^{N} \min_{j} \{d(\mathbf{x}_i, \mathbf{q}_j)\}$$

where $d(\mathbf{x}, \mathbf{q})$ is a measure of the distance (such as $(\mathbf{x}-\mathbf{q})^T \mathbf{C}^{-1} (\mathbf{x}-\mathbf{q})$ for the Mahalanobis distance where $\mathbf{C}$ is the covariance of the training data) between points $\mathbf{x}$ and $\mathbf{q}$. One way of estimating these points is called K-mean clustering, which groups training data into N clusters as follows.

1 **Initialise:** Choose N initial points randomly from the training set to use as initial estimates for the cluster means $\mathbf{q}_j$.

2 **Start Loop:** For each training point, calculate the cluster mean which is nearest to it in a weighted Euclidean sense. All of the training points which are closest to $\mathbf{q}_j$ are then assigned to the jth cluster.

3 Calculate the new cluster means $\mathbf{q}'_j$ by finding the mean of the points in the jth cluster.

4 **End Loop:** If $\mathbf{q}_j = \mathbf{q}'_j$ for all j, then end the loop, otherwise set $\mathbf{q}_j = \mathbf{q}'_j$ and return to step 2.

The loop termination condition in the above algorithm is practically rarely met, so for most practical applications alternative stopping criteria are required. Some methods use the change in the distortion measure or the total distortion to determine when to halt the loop, but due to the fast convergence properties of the method, just using a fixed number of iterations often works well. Some post-processing is also often necessary to remove clusters that may have formed containing either one or zero training points, since these are usually a result of overfitting.

Once a reduced set of points has been determined from the training data, the cluster means $\mathbf{q}_j$ can then be used in place of the original training set in a K-nearest neighbour discriminant (or in fact any other discriminant). Vector quantisation methods such as K-mean clustering can be enhanced for a particular discriminant through a technique referred to as Learning Vector Quantisation (LVQ) which is often associated with neural networks.

While VQ can be very useful, it should be noted that none of the possible VQ methods (short of a time-consuming combinatorial search) will consistently converge to the global minimum for the distortion. It also adds a model parameter to the discrimination procedure, since the optimal number of clusters to use is unknown. It does have the advantage for a K nearest neighbour discriminant that the size of the training set is effectively reduced, so the computation cost during the testing phase of the classifier will also reduced.

5.3.3 Neural Networks

Neural networks are connections of simple functional units called neurons, each of which contains various parameters or weights. These weights are usually controlled by the network itself in such a way that an output of the network will more closely approach a desired output for a given set of training vectors. In this way, the neural network can "learn" the expected output, and once the values of the weights have reached an equilibrium, the network can then be used to determine the expected output for any given input.

There are a great number of neural network architectures described in the literature, and it would be futile to review all of these. For this report, only the most popular method, the Multi-Layered Perceptron (MLP), will be discussed since it shares many important characteristics with other neural network methods.

For an MLP, an individual neuron is defined to produce a response dependent on the weighted sums of its inputs and a bias $\boldsymbol{\theta}$, as shown in Figure 5.1. The function g may be

any of a number of types, but the most commonly used are the sigmoid functions. The neurons are then interconnected to form a number of layers, as shown in Figure 5.2. As the size of this network tends to infinity, it can be shown that for sigmoid based neurons, any non-singular multidimensional function can be approximated to arbitrary precision. This indicates that finite sized neural networks can approximate a wide variety of functions.

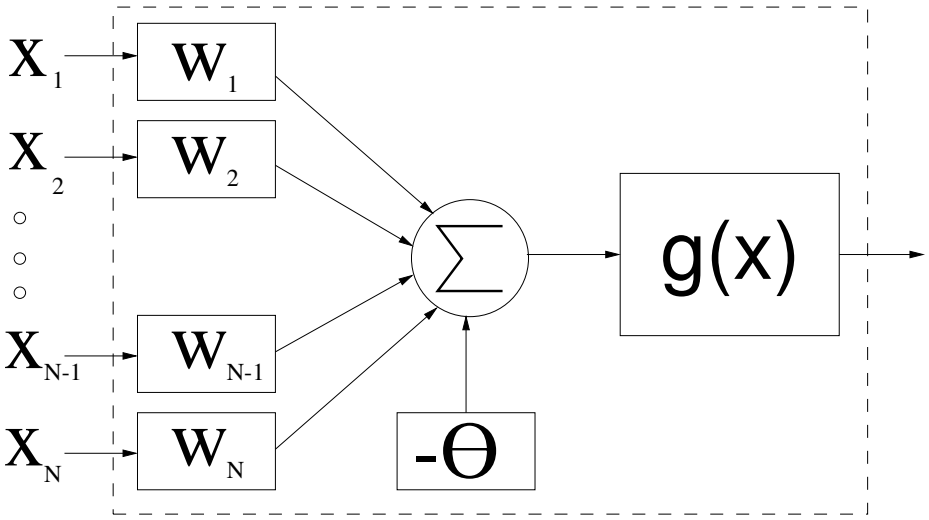


Figure 5.1: *Block diagram of a single neuron for an MLP*

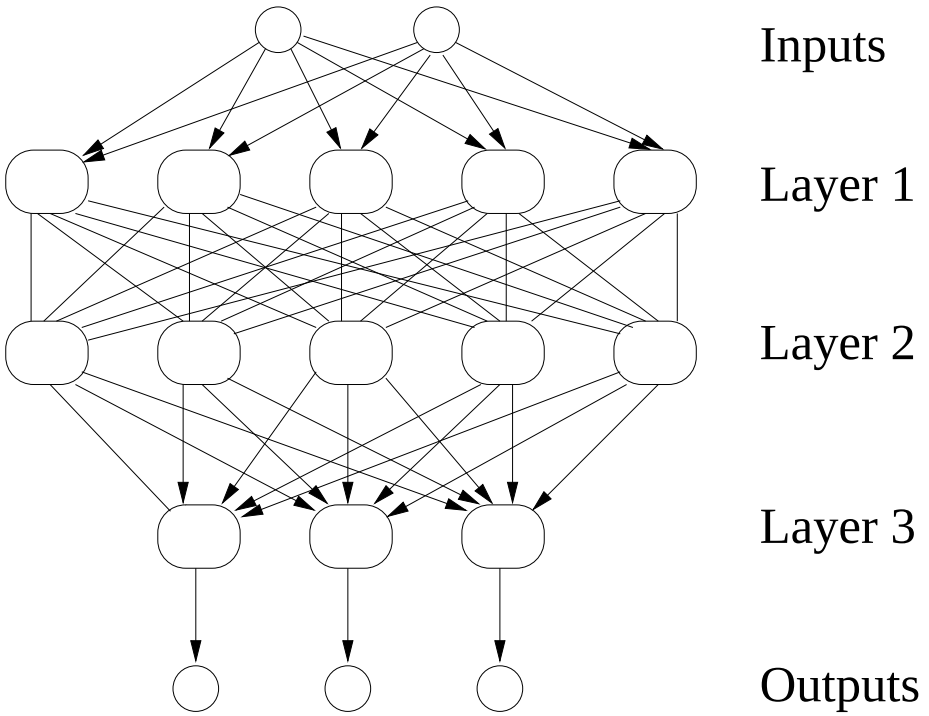


Figure 5.2: *An example of connections for a 3 level MLP*

The MLP may be used for two class discrimination of N points $\mathbf{x}_i$ for $i \in \{1, 2, ..., N\}$ into classes $y_i \in \{0, 1\}$, by solving the problem

$$\text{Minimise: } \sum_{i=1}^{N} C(f(\mathbf{w}, \boldsymbol{\theta}, \mathbf{x}_i), y_i).$$

Here C is the cost function associated with the neural network (having weights $\mathbf{w}$ and thresholds $\boldsymbol{\theta}$), producing an output of f for an input vector $\mathbf{x}_i$ instead of the desired output y_i. The weights $\mathbf{w}$ and thresholds $\boldsymbol{\theta}$ can be updated using steepest descent, or a second order method such as conjugate gradient to estimate the minimum. Once a minimum has been found in $\mathbf{w}$ and $\boldsymbol{\theta}$, the output of the neural network $f(\mathbf{w}, \boldsymbol{\theta}, \mathbf{x})$ can then be used to classify individual points.

Although MLPs can produce very good discrimination, they have a number of drawbacks which are common to most, if not all neural network schemes. Firstly, many parameters are required to be chosen such as the number of neurons in each layer, the number of layers, the learning rate (which is a parameter of the steepest descent method for finding the weights) and the functional form of the individual neurons. Again, these can only be determined by examining the error on a separate test set. Secondly, since the back-propagation method for determining the network weights is based on the principle of steepest descent, there is no guarantee that it will converge to the optimum, especially for large numbers of unknown weights. To improve the likelihood of finding a global minimum to the cost function, techniques such as simulated annealing can be used. These techniques effectively work by running the neural network from different initial weights and choosing the solution which gives the minimum cost. As a result, the training time is greatly increased still without the guarantee of an optimal solution.

5.3.4 Mercer Kernels

The standard technique for allowing SVMs, which are principally linear discriminants, to solve non-linear discrimination problems involves the use of Mercer kernels. Mika et.al. [9] shows that the same procedure can also be applied to the Fisher discriminant, yielding a fast non-linear discriminant (which is referred to as the Kernel Fisher Discriminant) having accuracy comparable to SVMs. In fact, the same technique can be applied to any linear discriminant whose functional form depends on dot products.

The basic idea behind the kernel method is that a non-linear decision surface can be exactly the same as a linear decision surface in a higher dimensional space. For instance, a quadratic discriminant in coordinates (x_1, x_2) can be obtained by constructing a linear discriminant in the five dimensional space having coordinates $(x_1, x_2, x_1^2, x_1x_2, x_2^2)$. For higher order discriminants however, the number of features required quickly becomes unmanageable. Suppose however that $\mathbf{x}$ is a point in the lower dimensional space, and $\boldsymbol{\Phi}(\mathbf{x})$ is a mapping of this point into a higher dimensional space. Then by using Mercer kernels, which are a set of functions $k(\mathbf{x}, \mathbf{y}) = \boldsymbol{\Phi}(\mathbf{x}).\boldsymbol{\Phi}(\mathbf{y})$ which express the dot product of the higher dimensional space in terms of the lower dimensional coordinates, it is often not necessary to perform the mapping $\boldsymbol{\Phi}$ directly. Two commonly used Mercer

kernels are the polynomial kernel $k(\mathbf{x}, \mathbf{y}) = (\mathbf{x}.\mathbf{y})^c$ or the Gaussian radial basis function $k(\mathbf{x}, \mathbf{y}) = \exp(-|\mathbf{x} - \mathbf{y}|^2/c)$ for some positive constant c.

The dual formulation of the linear SVM given in section 5.2.5 can be easily generalised to produce a non-linear discriminant using these Mercer kernels. The new optimisation problem to be solved can be written as

$$\begin{aligned} \text{Maximise:} \quad & \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{N} \alpha_i y_i \alpha_j y_j k(\mathbf{x}_i, \mathbf{x}_j) \\ \text{Subject to:} \quad & \sum_{i=1}^{N} \alpha_i y_i = 0, \ 0 \leq \alpha_i \leq C. \end{aligned}$$

for a particular kernel function $k(\mathbf{x}, \mathbf{y})$. This new problem will have exactly the same order of magnitude computational complexity as the linear version of the problem.

The Fisher linear discriminant can also be extended to take advantage of Mercer kernels. Following the analysis in Mika et. al. [9], since the normal to the decision surface between two distributions should belong to the vector space spanned by the points in the distributions, then we can write the normal vector of the linear discriminant in the higher dimensional space as

$$\mathbf{n} = \sum_{i=1}^{N} \alpha_i \mathbf{\Phi}(\mathbf{x}_i) \tag{5}$$

where $N = N_1 + N_2$ is the total number of points in both classes, and $\mathbf{x}_i$ is the ith point from the set of all points. Now if the two classes in the high dimensional space have means μ_1, μ_2 and covariances $\mathbf{C}_1$ and $\mathbf{C}_2$, then the Fisher discriminant maximises the expression for the separability given by

$$S = \frac{(\mathbf{n}^T(\mu_2 - \mu_1))^2}{\mathbf{n}^T\mathbf{C}_1\mathbf{n} + \mathbf{n}^T\mathbf{C}_2\mathbf{n}}.$$

In order to evaluate S without the need to evaluate the mapping $\mathbf{\Phi}$, equation (5) is applied to the expression containing the mean, yielding

$$\begin{aligned} \mathbf{n}^T\mu_1 &= \frac{1}{N_1}\mathbf{n}^T \sum_{j=1}^{N_1} \mathbf{\Phi}(\mathbf{x}_j^1) \\ &= \frac{1}{N_1} \sum_{i=1}^{N} \alpha_i \sum_{j=1}^{N_1} \mathbf{\Phi}(\mathbf{x}_i)^T \mathbf{\Phi}(\mathbf{x}_j^1) \\ &= \frac{1}{N_1} \sum_{i=1}^{N} \alpha_i \sum_{j=1}^{N_1} k(\mathbf{x}_i, \mathbf{x}_j^1) \\ &= \frac{1}{N_1} \alpha^{\mathbf{T}} \mathbf{k_1} \end{aligned}$$

where $\mathbf{x}_j^1$ is the jth point of the first class. In a similar way, the expression containing the covariance may be written after some working as

$$\begin{aligned}\mathbf{n}^T\mathbf{C}_1\mathbf{n} &= \frac{1}{N_1}\sum_{i=1}^{N_1}\left(\sum_{j=1}^{N}\alpha_j k(\mathbf{x}_i^1,\mathbf{x}_j)\right)^2 - (\mathbf{n}^T\mu_1)^2 \\ &= \frac{1}{N_1}\alpha^{\mathbf{T}}\mathbf{K_1}\mathbf{K_1^T}\alpha - \frac{1}{N_1^2}\alpha^{\mathbf{T}}\mathbf{k_1}\mathbf{k_1^T}\alpha\end{aligned}$$

The above expressions imply that the separability criterion to be maximised for the Fisher discriminant can be written as $S = (\alpha^{\mathbf{T}}\mathbf{A}\alpha)/(\alpha^{\mathbf{T}}\mathbf{B}\alpha)$ for some $N \times N$ matrices $\mathbf{A}$ and $\mathbf{B}$. The separability can be maximised by choosing α to be the eigenvector of $\mathbf{AB^{-1}}$ having the highest eigenvalue. Once α is known, the normal to the decision hyperplane in the higher dimensional space can be calculated from equation (5), and a ROC curve can be drawn by examining the distance of each of the distributions from the hyperplane passing through the origin. For a point $\mathbf{\Phi}(\mathbf{x})$, this distance can be calculated using

$$\mathbf{n}\mathbf{\Phi}(\mathbf{x}) = \sum_{i=1}^{N}\alpha_i k(\mathbf{x},\mathbf{x}_i).$$

which means that to classify a point, it is necessary to calculate N' (the number of support vectors) values of the kernel function. For a very large training set, this may become very large and the testing speed will be similar to that of the K-nearest neighbour method. As with that method, techniques such as vector quantization of the training set can be used to improve testing time, and for polynomial kernels much greater computational savings can be made by expansion and simplification of the above expression [13].

The 1D parameter search and recursive Fisher algorithms described earlier, can be implemented in a similar fashion. There is a slight difference however for the 1D parameter search, since it requires the evaluation of $(\mathbf{C}_1 + \gamma\mathbf{C}_2)^{-1}(\mu_2 - \mu_1)$ for the calculation of the search directions. This is the equivalent of maximising a separability of $S(\gamma) = (\mathbf{n}^T(\mu_2 - \mu_1))^2/(\mathbf{n}^T\mathbf{C}_1\mathbf{n} + \gamma\mathbf{n}^T\mathbf{C}_2\mathbf{n})$, or finding the eigenvector of $\mathbf{AB}(\gamma)^{-\mathbf{1}}$ having the highest eigenvalue, for an easily computable matrix $\mathbf{B}(\gamma)$.

Unlike the extension of the SVM, the new kernel Fisher based discriminants no longer have the same computational complexity as its original linear versions. For a set of N points, the Kernel Fisher methods require the inversion of an $N \times N$ matrix which is $\mathcal{O}(N^3)$ operations. This is comparable with the non-linear SVM. The kernel Fisher methods no longer have such an overwhelming computational advantage. Also, like all non-linear methods, the parameters of the kernel must be chosen and this must be done with the use of a training and validation set. Unlike the SVM however, it does not require that a regularisation parameter be chosen, which simplifies the training somewhat. Also unlike the SVM, the results are scale independent (at least for scale independent kernels like polynomials) which should make the discriminant slightly more robust. On the whole however, the SVM and Kernel Fisher methods are similar performance wise [9].

5.4 Discrimination enhancements

5.4.1 Bagging and Boosting

Boosting and bagging [10] are two examples of voting based methods. These rely on the calculation of a number of discriminants and use the classification produced by all of these in a voting system to decide the class of a particular point.

Bagging (also known as bootstrap aggregating) can be applied to a training set of size N in the following way.

1 Generate a series of k training sets of size N from the original set. This is done randomly with replacement (so even though the training sets have the same size as the original, some of the points may be repeated).

2 Apply a discrimination algorithm of some kind to each of the training sets to produce k decision surfaces.

3 To classify a point, count the votes for each class of the k individual discriminants. The class which obtains the most votes is taken as being the class of that point.

This technique can in some circumstances produce a much more robust and predictive classifier than each of the individual discriminants. Many classifiers however such as the Fisher linear discriminant, are quite stable to perturbations of the training set for reasonably sized data sets. As a result, the k different discriminants which are used for voting will be reasonably similar and so there will be little or no improvement in performance. On the other hand, for extremely high dimensional data, or for many non-linear discriminants which can overfit to a training set, bagging can give considerable performance improvement.

Boosting of a discriminant is similar to bagging except that each of the voting classifiers is given a weight which depends on its successfulness. There are a number of ways in which boosting can be achieved for a discriminant on a training set of size N, and one of these termed AdaBoost.M1 is as follows.

1 **Initialisation:** Each point $\mathbf{x}_j$ of the training set is given a weight of $w_j = 1/N$. i is set equal to 1.

2 **Loop:** Some discriminant method is used on the training set with weights $\mathbf{w}$ to produce classifier i.

3 **Weighting the classifier:** The error on the training set ϵ is then calculated by summing the weights of the misclassified points. If this error is greater than 50 percent, then the loop stops and the complete classifier is based only on the previously obtained discriminants. Otherwise the correctly classified points are weighted less by multiplying their weights by $\beta_i = \epsilon/(1-\epsilon)$ and then all of the weights are rescaled so that they sum to 1.

4 **End of loop:** Goto step 2 unless the number of required discriminants for boosting has been reached.

5 **Classification:** A particular point is classified by a vote using all of the calculated discriminants, where the ith classifier is weighted by a factor $\log(1/\beta_i)$.

The numerical experiments performed in [10] suggest that boosting generally produces better results than bagging. It should be noted however that when the discriminant which is being boosted overfits the data so that all training points are correctly classified, boosting will not improve the test error. Quinlan [10] also observes that boosting may sometimes cause the test error to increase, and suggests this may be due to overfitting. As a result, although boosting may result in a better classifier than bagging, it requires greater care in its use.

5.5 Conclusion

In this report, a number of discriminant methods have been outlined with some brief explanations of their advantages and disadvantages. While there are many other types that have not been covered, they are mostly refinements of the types mentioned here.

For the JP129 project, discriminants are required in the low level classification stage for distinguishing targets from background. While probably the most important aspect for the final performance of this stage is the type of 'features' to use in classification, the classifier also plays an important part. For instance, in order to determine whether a feature is useful, a discriminant needs to be used to see how much of the target distribution is separated from the background by the feature's inclusion. In this case, a poor classifier will be unable to give a good estimate of this, resulting in the discarding of a potentially useful feature. Also, a classifier of some sort has to be used in the final system, and any performance improvement available here will strongly affect overall performance.

The feature selection component of the system design chooses features in a recurrent pair-wise manner as described in [3]. The choice of classifier for this stage must therefore take into consideration two main points. Firstly, the features giving the best discrimination for one classifier may not be those which have the best separation overall. As a result, the selected features will be biased towards the particular discriminant used in selection. By far the most important point however is that during feature selection, it is necessary to run thousands of classifications on hundreds of features. Due to the overwhelming amount of computation necessary, the best discriminant to use must be fast and require a minimal amount of human intervention. For this reason, linear discriminants are really the only candidates for this purpose, and it was found that the recursive Fisher algorithm, which was specifically developed for this project, was the best of the tested alternatives.

Once the features for the final classifier have been selected, a discriminant which gives closer to optimal separation can then be used to squeeze extra performance from the final system. Since this classifier only needs to be trained once, more time consuming methods requiring more human supervision can be used. It is suggested that some sort of non-linear discriminant be used. Since the distributions to be separated appear to be unimodal, a

Gaussian mixture model is possibly not appropriate, and it is suggested that either a kernel linear discriminant (such as a support vector machine) or a K-nearest neighbour type discriminant be used for the final classifier. One thing that must be kept in mind however is the testing time per image, since this may adversely affect the total speed of the system.

Both the K-nearest neighbour and SVM classifiers are quite slow for testing, and will take a time roughly proportional to the number of training vectors to classify a particular image. Using vector quantisation may reduce this time, but an even greater saving could be made by using a polynomial kernel linear discriminant. Of course these computational savings will affect the performance of the classifier, but these trade-offs are unavoidable.

Another technique for reducing the computation time is to cascade classifiers. By splitting the low level classification into two stages where the first stage is a very high PD, very fast, linear classifier, the number of false alarms can be reduced for the second stage. Hence a more time consuming classifier can be used without any effect on the speed of the complete system.

References

1. Anderson, T.W. and Bahadur, R.R., "Classification into two multivariate normal distributions with different covariance matrices", Annals of Mathematical Statistics, Vol.33, June 1962, pp.420-431.

2. Burges C.J.C., "A tutorial on support vector machines for pattern recognition," Data Mining and Knowledge Discovery, Vol.2, No.2, pp.1-47, 1998.

3. Cooke T.P., First Report on Features for Target/Background Classification, CSSIP-CR-9/99.

4. Cooke T.P., Second Report on Features for Target/Background Classification, CSSIP-CR-26/99.

5. Cooke T.P. and Peake M., "The optimal classification using a linear discriminant for two point classes having known mean and covariance," Submitted to the Journal of Multivariate Analysis.

6. Cover T.M. and Hart P.E., "Nearest neighbour pattern classification," IEEE Transactions on Information Theory, Vol.3, 1967, pp.21-27.

7. Fisher R.A., "The use of multiple measurements in taxonomic problems," Annals of Eugenics, Vol.7, Part II, pp.179-188, 1936.

8. Jarrad G.A. and McMichael D.W., "Shared mixture distributions and shared mixture classifiers," Procedings of Information Decision and Control 99.

9. Mika S., Rätsch G., Weston J., Schölkopf B. and Müller K., "Fisher discriminant analysis with kernels," to appear in 1999 IEEE Workshop on Neural Networks for Signal Processing IX.

10. Quinlan J.R., "Bagging, Boosting and C4.5," Procedings of the 13th American Association for Artificial Intelligence, pp.725-730, AAAI Press, Menlo Park CA, 1996.

11. Redding N.J., Design of the Analysts' Detection Support System for Broad Area Aerial Surveillance, DSTO-TR-0746.

12. Robinson D.J. and Redding N.J., Prescreening Algorithm Performance in the Analyst's Detection Support System, DSTO-RR-0000.

13. Tang D., Schroder J., Redding N.J., Cooke T., Zhang J. and Crisp D., "Computationally efficient classification with support vector machines by kernel decomposition", submitted to VC2000.

Chapter 6

Prescreeners

6.1 Introduction

This report describes the research into target detection/recognition in maritime surveillance SAR imagery. The major focus of this first report is the prescreener, which is the algorithm which scans large amounts of imagery quickly to winnow the data down to a relatively small number of candidate targets. Some theoretical and numerical results have been obtained for a number of prescreeners, and these have been described in Section 6.2. The next stage in the SAR image processing chain is the low level classifier, which uses more computationally intensive algorithms to check each of the candidate targets, and remove a large fraction of the false alarms. Section 6.3 describes a few low level classification algorithms which were briefly looked at during the current contract. A more detailed look at the low level classifier will be forthcoming in a subsequent report.

6.2 Prescreening

Most adaptive prescreeners are based on a statistical comparison between a group of pixels, hypothesised to belong to the target, and a collection of points belonging to the background. Perhaps the most commonly referred to prescreener is that used by Lincoln Laboratory for automated target detection [11]. This uses a rectangular template region centered on a single pixel of interest (referred to here as the signal window. See Figure 6.1). The pixels at the edge of the rectangle are assumed to correspond to background clutter, and the remaining pixels form a guard ring around the central pixel. The pixels from the guard ring may contain pixels corresponding to a hypothetical target at the centre, so are not used in the estimation of the background statistics. The prescreener then calculates a threshold, based on the statistics of the background pixels, so that the percentage of background pixels above the threshold should be constant. By detecting all central pixels above this adaptive threshold, a detector with a Constant False Alarm Rate (CFAR) is produced.

There are many modifications of the above CFAR detector discussed in the literature [1]. For instance, different models for the statistics of background scatterer intensities may be used (Gaussian, Weibull, K-distribution, etc). Even for the specific assumption

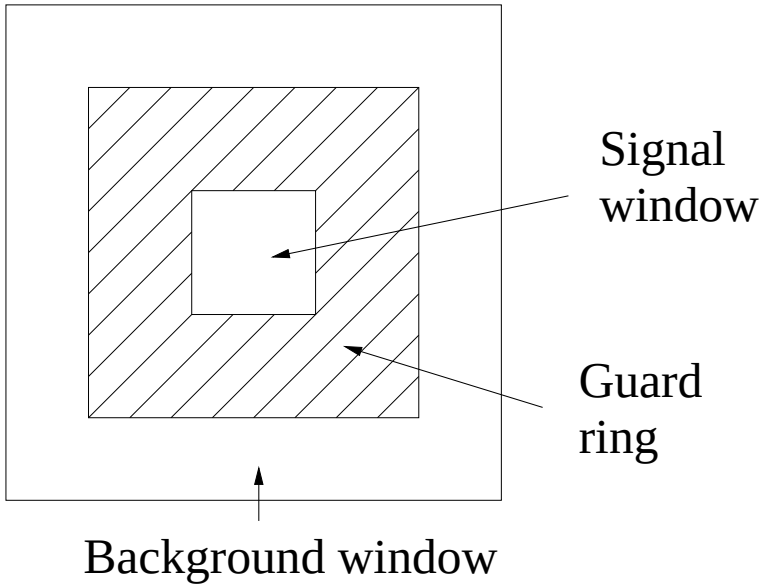


Figure 6.1: *The template for the Lincoln Laboratory CFAR detector*

of K-distributed clutter, there are many different ways of estimating the CFAR threshold parameters from the background pixel samples. Furthermore, alterations in the template geometry, including changes in the shape of the background window, and varying the size of the central signal window have also been discussed.

This section describes a number of variations on the standard CFAR detector theme. Subsection 6.2.1 describes a prescreener based on a specific distribution (the G distribution), while the remaining parts deal with non-parametric detectors which aim to avoid errors due to incorrect modelling of the underlying pixel statistics. Subsection 6.2.2 uses a non-parametric estimate to determine the rate of decay of the tail distribution, and uses this to compute a threshold. Subsections 6.2.3 and 6.2.4 also describe a non-parametric CFAR detector based on the worst possible false alarm rate for a given background mean and variance. This worst case scenario detector has also been used to provide a theoretical measure of the effect of ship size and radar resolution on prescreener performance. Finally, subsection 6.2.5 describes a template based prescreener which accounts not only for the distribution of background pixels, but also the distribution of target pixels. A brief comparison of these detectors is given in subsection 6.2.6.

6.2.1 The G distribution

A previous report [8], describes several types of statistical distributions which have been used, with varying degrees of success, for modelling background clutter distributions. The simple Gaussian case is used by Lincoln Labs in their single point CFAR detector [11] for land targets, as well as by Wackerman [17] for maritime targets. K-distribution based CFAR detectors have also been considered for use in INGARA imagery, and several reports describe how the background parameters should be estimated [14], how the

prescreener is incorporated into the ADSS system [12], and a comparison of prescreener results [2]. Recently, a number of papers have been using the G distribution, and related approximations such as the G^0 distribution which becomes the β' distribution [15] for the special case where the radar imagery averages only one look. Salazar gives the G^0 distribution as

$$f_n(x,\alpha,\gamma) = \frac{\Gamma(n+\alpha)(\gamma/n)^\alpha}{\Gamma(n)\Gamma(\alpha)} \frac{x^{n-1}}{(x+\gamma/n)^{n+\alpha}},$$

where n is the number of looks, and α and γ are two positive parameters of the distribution. For the special case when $n = 1$, Salazar gives the mean and variance of the distribution to be

$$\mu = \frac{\gamma}{\alpha-1} \tag{1}$$

$$\sigma^2 = \frac{\alpha\mu^2}{\alpha-2} \tag{2}$$

so that the parameters of the distribution may be easily obtained from estimates of the first two moments. These parameter estimates will not be the maximum likelihood estimates, but for sufficient numbers of samples, should provide a fast method for the parameter estimation. When $n > 1$, the mean may be calculated as follows:

$$\begin{aligned}
\mu_{n+1}(\alpha,\gamma) &= A_{n+1}(\alpha,\gamma)\int_0^\infty \frac{x^n}{(x+\frac{\gamma}{n+1})^{n+\alpha+1}} x dx \\
&= A_{n+1}(\alpha,\gamma)\left[\int_0^\infty \frac{x^n}{(x+\frac{\gamma}{n+1})^{n+\alpha}} - \frac{\frac{\gamma}{n+1}x^n}{(x+\frac{\gamma}{n+1})^{n+\alpha+1}} dx\right] \\
&= A_{n+1}(\alpha,\gamma)\int_0^\infty \frac{x^{n-1}}{(x+\frac{n}{n+1}\gamma/n)^{n+\alpha}} x dx - \frac{\gamma}{n+1} \\
&= \frac{A_{n+1}(\alpha,\gamma)}{A_n(\alpha,\frac{n}{n+1}\gamma)}\ \mu_n\left(\alpha,\frac{n}{n+1}\gamma\right) - \frac{\gamma}{n+1} \\
&= \frac{n+\alpha}{n}\ \mu_n\left(\alpha,\frac{n}{n+1}\gamma\right) - \frac{\gamma}{n+1}
\end{aligned}$$

Now we hypothesise that $\mu_n(\alpha,\gamma) = \mu_1(\alpha,\gamma)$, so substituting into this expression gives

$$\begin{aligned}
\mu_{n+1}(\alpha,\gamma) &= \frac{n+\alpha}{n}\frac{n}{n+1}\frac{\gamma}{\alpha-1} - \frac{\gamma}{n+1} \\
&= \frac{\gamma}{n+1}\left(\frac{n+\alpha}{\alpha-1} - 1\right) \\
&= \frac{\gamma}{\alpha-1}
\end{aligned}$$

and so by induction, the hypothesis holds true in general. A similar argument can be made for the variance, which is given by

$$\begin{aligned}
\sigma^2_{n+1}(\alpha,\gamma)+\mu^2_{n+1}(\alpha,\gamma) &= A_{n+1}(\alpha,\gamma)\int_0^\infty \frac{x^n}{(x+\frac{\gamma}{n+1})^{n+\alpha+1}}x^2dx \\
&= A_{n+1}(\alpha,\gamma)\left[\int_0^\infty \frac{x^{n-1}}{(x+\frac{\gamma}{n+1})^{n+\alpha}}x^2 \right. \\
&\qquad \left. -\frac{\frac{\gamma}{n+1}x^n}{(x+\frac{\gamma}{n+1})^{n+\alpha+1}}xdx\right] \\
&= \frac{A_{n+1}(\alpha,\gamma)}{A_n(\alpha,\frac{n}{n+1}\gamma)}\left[\sigma^2_n(\alpha,\frac{n}{n+1}\gamma)+\mu^2_n(\alpha,\frac{n}{n+1}\gamma)\right] \\
&\quad -\frac{\gamma}{n+1}\mu_{n+1}(\alpha,\gamma) \\
&= \frac{n+\alpha}{n}\left[\sigma^2_n(\alpha,\frac{n}{n+1}\gamma)+\mu^2_n(\alpha,\frac{n}{n+1}\gamma)\right] \\
&\quad -\frac{\gamma^2}{(n+1)(\alpha-1)}
\end{aligned}$$

It is hypothesised that $\sigma^2_n(\alpha,\gamma)+\mu^2_n(\alpha,\gamma) = (1+1/n)\gamma^2/((\alpha-1)(\alpha-2))$, so substituting into the above equation gives

$$\begin{aligned}
\sigma^2_{n+1}(\alpha,\gamma)+\mu^2_{n+1}(\alpha,\gamma) &= \frac{n+\alpha}{n}\frac{n\gamma^2}{(n+1)(\alpha-1)(\alpha-2)}-\frac{\gamma^2}{(n+1)(\alpha-1)} \\
&= \frac{n+2}{n+1}\frac{\gamma^2}{(\alpha-1)(\alpha-2)}
\end{aligned}$$

so again by induction, the formula holds true in general. This means that given the number of looks n, the parameters α and γ may be estimated from the moments estimates using the formulae

$$\hat{\alpha} = 1+n\frac{\hat{\sigma}^2+\hat{\mu}^2}{n\hat{\sigma^2}-\hat{\mu}^2} \tag{3}$$

$$\hat{\gamma} = (\hat{\alpha}-1)\hat{\mu} \tag{4}$$

If n is not known in advance, a minimum value for n may be estimated by noting that the denominator of equation (3) must be positive in order that $\hat{\gamma}$ stays positive. Therefore $n \geq (\hat{\mu}/\hat{\sigma})^2$.

An expression for the cumulative distribution can be derived in a similar way, but does not seem to give a nice analytical form. For n looks, the probability that x is larger than some threshold x_0 is

$$t_{n+1}(x_0,\alpha,\gamma) = \int_{x_0}^\infty A_{n+1}(\alpha,\gamma)\frac{x^n}{(x+\frac{\gamma}{n+1})^{n+\alpha+1}}dx$$

$$
\begin{aligned}
&= A_{n+1}(\alpha,\gamma)\left\{\left[-\frac{x^n}{(x+\frac{\gamma}{n+1})^{n+\alpha}(n+\alpha)}\right]_{x_0}^{\infty}\right. \\
&\qquad \left. +\int_{x_0}^{\infty}\frac{nx^{n-1}}{(x+\frac{\gamma}{n+1})^{n+\alpha}(n+\alpha)}dx\right\} \\
&= A_{n+1}(\alpha,\gamma)\left\{\frac{x_0^n}{(x_0+\frac{\gamma}{n+1})^{n+\alpha}(n+\alpha)}\right. \\
&\qquad \left. +\frac{n}{n+\alpha}\int_{x_0}^{\infty}\frac{x^{n-1}}{(x+\frac{\gamma}{n+1})^{n+\alpha}}\right\} \\
&= A_{n+1}(\alpha,\gamma)\frac{x_0^n}{(x_0+\frac{\gamma}{n+1})^{n+\alpha}(n+\alpha)} \\
&\quad +\frac{n}{n+\alpha}\frac{A_{n+1}(\alpha,\gamma)}{A_n(\alpha,\frac{\gamma n}{n+1})}\,t_n(\alpha,\frac{\gamma n}{n+1}) \\
&= A_{n+1}(\alpha,\gamma)\frac{x_0^n}{(x_0+\frac{\gamma}{n+1})^{n+\alpha}(n+\alpha)}+t_n(x_0,\alpha,\frac{\gamma n}{n+1})
\end{aligned}
$$

The solution to this equation will be given by

$$
t_n(x_0,\alpha,\gamma)=\left(\frac{\gamma/n}{x_0+\gamma/n}\right)^{\alpha}\left\{\sum_{i=0}^{n-1}\frac{\Gamma(\alpha+i)}{\Gamma(\alpha)i!}\left[\frac{x_0}{x_0+\gamma/n}\right]^i\right\}
$$

which, for a given false alarm rate, should be solved for x_0 numerically. This value for x_0 can be used as a threshold in a CFAR detector based on the G^0 distribution for the background clutter.

6.2.2 Hill's estimator

The exact density function for background pixel intensities is not known, although some good empirical models (such as the G distribution from the previous subsection) are available. Due to the low false alarm rates desired from prescreeners however, it is only really important to have good estimates for the tail of the distribution. Empirically fitting models such as the K or G distributions may reduce the accuracy of the tail estimate while improving the overall discrepancy between the data and the distribution. As a result, it is of interest to fit the tail separately from the rest of the distribution.

Hill's estimator [10] is an order statistic based non-parametric quantity used to measure the asymptotic behaviour of a probability density function. Suppose a random variable X has a cumulative density function $F(x)$ with a fat tail, defined by

$$
x \overset{\lim}{\to} \infty\; x^{-1/\gamma}(1-F(x))=C
$$

for some finite C. The number $1/\gamma$ is referred to as the tail index of the distribution. Further suppose that ordered samples of the random variable X_i are available where $X_i > X_{i+1}\ \forall i < N$. Then Hill's estimate for γ is given by

$$\hat{\gamma}(k) = \frac{1}{k}\sum_{i=1}^{k} \log\left(\frac{X_i}{X_{k+1}}\right).$$

The required threshold for a given false alarm rate can then be estimated in the following way:

- **Find the start of the tail:** Plot cumulative distribution functions for both the estimated tail, and the sample points. The start of the tail x will roughly occur when the sample cdf deviates from the tail pdf. This can be determined by using a hypothesis test based on a threshold of the Kolmogorov-Smirnov statistic. As a simpler and faster alternative however, the results in subsection 6.2.6 assume the tail begins in the top 10 percent of pixel intensities.

- **Find the area of the tail:** If f is the fraction of pixels contained in the tail of the distribution (*i.e.* brighter than x), then

$$C\int_{t=x}^{\infty} x^{-1/\hat{\gamma}}dx = C\frac{1}{1/\hat{\gamma}-1}x^{1-1/\hat{\gamma}} = f \tag{5}$$

  which can then be written as an expression for C.

- **Calculating threshold:** By substituting the desired false alarm rate for f into equation (5) as well as the value for C calculated previously, the required threshold can be calculated by solving for x.

Subsection 6.2.6 gives a brief comparison of this CFAR detector with some of the others to be described in this section.

6.2.3 Single point detector

The K-distribution is the most commonly used model for single-point statistics of sea clutter, but it is not perfect and there are a number of papers (for example [15]) describing more generalised distributions which give better empirical fit to the data. Instead of making incremental improvements to the clutter model, which may or may not particularly well fit the actual data for any particular radar configuration, or background surface, an alternative approach is to use a theoretical upper bound on the false alarm rate given some simple non-parametric assumptions about the distribution. Such a bound is expected to be more robust to variation than other distribution specific methods.

The original CFAR detector used a simple adaptive threshold $\mu + k\sigma$ where μ and σ are estimates of the local single point mean and covariance for the background. If it is assumed that there are sufficiently large numbers of background pixels (so that the parameter estimates are accurate) and that the single point distribution is unimodal, then one can show [3] that the percentage of false alarms will be bounded by

$$\begin{cases} 1 - \dfrac{4k^2}{3(1+k^2)} & \text{for } k \leq \sqrt{\dfrac{5}{3}} \\ \dfrac{4}{9(1+k^2)} & \text{otherwise.} \end{cases} \tag{6}$$

Most prescreeners will use a fairly low false alarm rate, so k is likely to be much greater than $\sqrt{5/3}$ and the second formula will apply. These formulae can be compared with similar curves obtained by making distributional assumptions about the background clutter statistics. For instance, if the clutter is Gaussian, the fraction of false alarms will be

$$\text{Gaussian FAR} = \frac{1}{2}\left(1 - \text{erf}\left(\frac{k}{\sqrt{2}}\right)\right) \sim \frac{1}{k}\exp\left(-\frac{k^2}{2}\right) \quad \text{as } k \to \infty.$$

and if the clutter is exponentially distributed, the fraction will be

$$\text{Exponential FAR} = \int_{k+1}^{\infty} \exp(-x)dx = \exp(-k-1).$$

Plots of the false alarm rate as a function of k are shown in Figure 6.2. The Gaussian and exponential plots (as well as for gamma and K-distributed variables) have previously been documented by Wackerman et. al. [17]. As the threshold is increased, the false alarms drop off much more quickly for the Gaussian than for the longer tailed distributions such as the exponential or K-distributions.

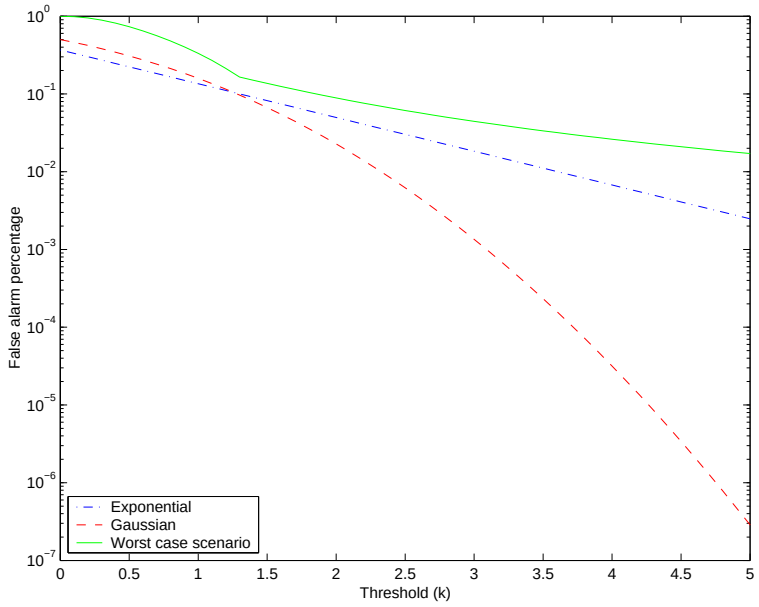


Figure 6.2: *Single pixel CFAR detector false alarm rates as a function of threshold*

The formulae for the graphs in Figure 6.2 can be used to quickly derive a very rough formula for the effect of signal window size on prescreener false alarm rate. In an N pixel prescreener, the mean of the pixels in the signal window is again adaptively compared to the background, with the detection threshold set to $\mu + k\sigma$ where μ and σ^2 are the single point statistics of the background.

Suppose that the target is of sufficient size such that every pixel in the signal window is a target pixel. Further suppose (for the lack of a better model) that the target pixel intensities are constant. For this model, changing the signal window size should not affect the probability of detection of the prescreener, so the prescreener performance may be measured by the false alarm rate alone. If the signal window contains N pixels, then whenever the signal window contains only background, the sample mean will have mean μ and variance σ^2/N. The sample mean is thus $\sqrt{N}k$ standard deviations below the threshold, so in the worst case the fraction of false alarms will be

$$\text{Fraction of false alarms} = \frac{4}{9(1+Nk^2)}. \tag{7}$$

Now when the resolution improves so that the number of pixels on the target is increased by a factor of N, the target window may also be increased by a factor of N. The statistics of the background pixels will also change in practice, depending on the correlation properties of the image, but for this example they are assumed to stay the same. Since the number of pixels that must be searched by the prescreener also increases by a factor of N, this gives a false alarm rate proportional to $(4/9)N/(1+Nk^2) \to 4/(9k^2)$ as $N \to \infty$. This makes the false alarm rate roughly independent of the radar resolution.

Figure 6.2 shows that the worst case error has a very slow rate of fall as k increases. If the background were assumed to be Gaussian instead, a similar calculation would show that the false alarm rate would fall quite rapidly with improved resolution. It is known from the central limit theorem that as more pixels are incorporated into the signal window, the distribution of the mean will be more normally distributed, so a stringent upper bound should also experience a fall with improved resolution. A more accurate bound on false alarm rate is discussed in subsection 6.2.4.

6.2.4 Multi-pixel target detection

The CFAR detector shown in Figure 6.1 which thresholds an image pixel based on an estimate of the background statistics, is the best that can be done using only single point statistics of a target. There are still some minor improvements that can be made on this basic design. The problem of the best way to choose the threshold seems to produce a lot of papers on different clutter distributions, but little noticeable improvement in system performance. Similarly, different sized and shaped background windows may improve estimates of the background statistics. For instance, pixels with the same range as the candidate target might be removed from the background estimate so that the Doppler blurring from a moving target will not interfere with the background estimate. Again, the improvement achievable through these modifications are likely to be minor.

In order to gain improvement in target detection, some extra information about the target itself must be used. A natural extension is to consider spatial correlation and

multi-point statistics of targets and background clutter, and the easiest way to do this is to increase the size of the target window in Figure 6.1 to n pixels. In this subsection, each of the target window pixels is given an equal weight, and the mean over the pixels is compared with the background. An alternative proposal with unequal weights is described in Subsection 6.2.5 on template matching.

Suppose a target is known to consist of at least n pixels connected in a particular known spatial arrangement corresponding to the distribution of pixels in the target window. Then a fixed threshold c on the sample mean over these pixels will result in a detector with a constant probability of detection. Suppose the background pixels are independent and identically distributed. Then it is of interest to determine, given a background mean μ_b and variance σ_b^2, an upper limit on the probability of a false alarm as a function of the number of pixels n.

As $N \to \infty$, the distribution of the mean of N background target pixels will become a normal distribution with mean μ_b and variance σ_b^2/N, by the Central Limit Theorem. One relevant theorem which describes how quickly the random sum becomes Gaussian with increasing n is the Berry-Essèen theorem which states that

$$\sup_z |P(Z_n \leq c) - \Phi(c)| \leq \frac{33}{4} \frac{E(|X|^3)}{\sqrt{n}},$$

where $\Phi(c)$ is the cumulative distribution function for a normalised Gaussian and Z_n is the sum of random variables which has been normalised to be zero mean and unit variance. Later results in the literature have concentrated on finding a lower limit on the 33/4 factor, and empirical studies have found that 2.05 seems to be achievable.

The Berry-Esséen theorem is not especially useful for this problem because it does not pinpoint the position along the cumulative distribution function where the distribution differs most from the normal approximation. Also, for certain pathological distributions (for which the third moment is infinite) it is completely useless since the bound is infinite. While some work exists in the literature relating to the type of bound required, no definitive result seems to exist for finite $n > 1$.

6.2.4.1 The two point case

Theorem 1: Suppose $f(x)$ is a pdf and $g(x)$ is an odd monotonically increasing function such that the support of f is a subset of the range of $g + c$. Then $P(X + X > 2c)$ for X $f(x)$ and $P(Y + Y > 2c)$ for Y $f(g(x - c) + c)g'(x - c)$ are identical.

Proof:

$$\begin{aligned} P(Y + Y > 2c) &= \int_{-\infty}^{\infty} P(Y > 2c - x) f(g(x - c) + c) g'(x - c) dx \\ &= \int_{-\infty}^{\infty} \int_{2c-x}^{\infty} f(g(\xi - c) + c) g'(\xi - c) d\xi \; f(g(x - c) + c) g'(x - c) dx \end{aligned}$$

$$= \int_{-\infty}^{\infty} \int_{c+g(c-x)}^{g_{max}+c} f(y)dy\ f(g(x-c)+c)g'(x-c)dx$$

where $(y-c) = g(\xi - c)$. Now using the fact that g is odd and the substitution $(z-c) = g(x-c)$ gives

$$\begin{aligned} P(Y+Y>2c) &= \int_{g_{min}+c}^{g_{max}+c} \int_{2c-z}^{g_{max}+c} f(y)dyf(z)dz \\ &= \int_{-\infty}^{\infty} \int_{2c-z}^{\infty} f(y)dyf(z)dz \\ &= P(X+X>2c). \end{aligned}$$

Empirical results: The above theorem means that if there exists a transformation $g(x)$ such that the new density function Y has mean $\mu_Y < \mu_X$ and $\sigma_Y^2 < \sigma_X^2$, then it is possible to generate a new density function with $P(Y+Y>2c) > P(X+X>2c)$. This means that the distribution function which gives an upper bound on this probability must either be invariant to the transformation, or such a transformation must not exist.

A gradient-descent based variational method can be used to obtain a locally optimal distribution function for maximising $P(X+X>2c)$. Consider the function $g(x) = x + \varepsilon h(x)$ in Theorem 1 for some small ε. Then the distribution $f(g(x-c)+c)g'(x-c)$ will have an identical version for $P(X+X>2c)$, but the mean will be given by

$$\begin{aligned} \mu &= \int_{-\infty}^{\infty} xf(x+\varepsilon h(x))(1+\varepsilon h'(x))dx \\ &= \int_{-\infty}^{\infty} x(f(x)+\varepsilon f'(x)h(x)+\mathcal{O}(\varepsilon^2))(1+\varepsilon h'(x))dx \\ &= \int_{-\infty}^{\infty} xf(x)+\varepsilon\left(xf'(x)h(x)+xf(x)h'(x)\right)dx \\ &\approx \varepsilon\int_{-\infty}^{\infty} x\frac{d}{dx}(f(x)h(x))dx, \end{aligned}$$

since the mean of the pdf $f(x)$ is known to be zero. After integration by parts (and using the assumption that the support is $f(x)$ is finite, and looking at the limit as the support tends to infinity, to avoid problems with unusual distributions),

$$\mu \approx -\varepsilon\int_{-\infty}^{\infty} f(x)h(x)dx \tag{8}$$

is obtained, where the function $h(x)$ may be chosen so that the right hand side is zero. Similarly, an expression for the variance of the new distribution can be written as

$$\sigma^2 = \int_{-\infty}^{\infty} x^2 f(x+\varepsilon h(x))(1+\varepsilon h'(x))dx$$

$$\begin{aligned} &\approx 1+\varepsilon\int_{-\infty}^{\infty} x^2\frac{d}{dx}(f(x)h(x))dx \\ &= 1-2\varepsilon\int_{-\infty}^{\infty} xf(x)h(x)dx. \end{aligned} \tag{9}$$

Now in order for $x+\varepsilon h(x)$ to still be a monotonic odd function about $x=c$, $h(x)$ also needs to be odd about $x=c$ (for ε small enough, the monotonic part shouldn't matter). The simplest possible form for the function $h(x)$ is a cubic polynomial of the form

$$h(x) = (x-c+\xi)(x-c)(x-c-\xi), \tag{10}$$

for some unknown constant $\xi > 0$. This now forms the basis of a variational method, consisting of the following steps:

- **Step 1:** Initialise the distribution $f(x)$ to have zero mean and unit variance. A normal distribution might be a good starting point, but any distribution will do.
- **Step 2:** Choose the function $h(x)$ such that the integral on the right hand side of equation (8) is equal to zero. This can be done by performing a $1D$ search for ξ in equation (10). If such a ξ does not exist, then stop the procedure.
- **Step 3:** For a fixed magnitude step size ε, choose the sign of the step such that the variance from equation (9) decreases. Then take a step in this direction by defining $\hat{f}(x) = f(x+\varepsilon h(x-c))(1+\varepsilon h'(x-c))$.
- **Step 4:** Renormalise $\hat{f}(x)$ so that it has zero mean and unit variance. This new distribution should give a probability $P(X+X>2c)$ at least as large as that of the original distribution.
- **Step 5:** Set the new value of f to $\hat{f}$, and then return to Step 2 until the algorithm converges.

The above algorithm was implemented in MATLAB, and initially seeded with a normal distribution. Figure 6.3 shows how the distribution converges to two δ-functions when $c=1$. While it could be possible that this is just a local optimum for the problem, similar distributions are obtained for a wide range of c, and for a number of different initial distribution types. This observation leads to the following hypothesis:

Hypothesis 1: If $f(x)$ is the distribution function of normalised i.i.d. random variables X_i which gives the largest value for $P(X_1+X_2>2c)$, then $f(x)$ is the sum of two delta functions, so to satisfy the mean and variance constraints must be of the form

$$f(x) = \frac{1}{1+a^2}\delta(x-a) + \frac{a^2}{1+a^2}\delta(x+1/a).$$

This means X_i is a binary value, with possible values of a and $-1/a$, so the sum X_1+X_2 will have contributions at $2a$, $a-1/a$ and $-2/a$. Obviously $a \geq c$, otherwise

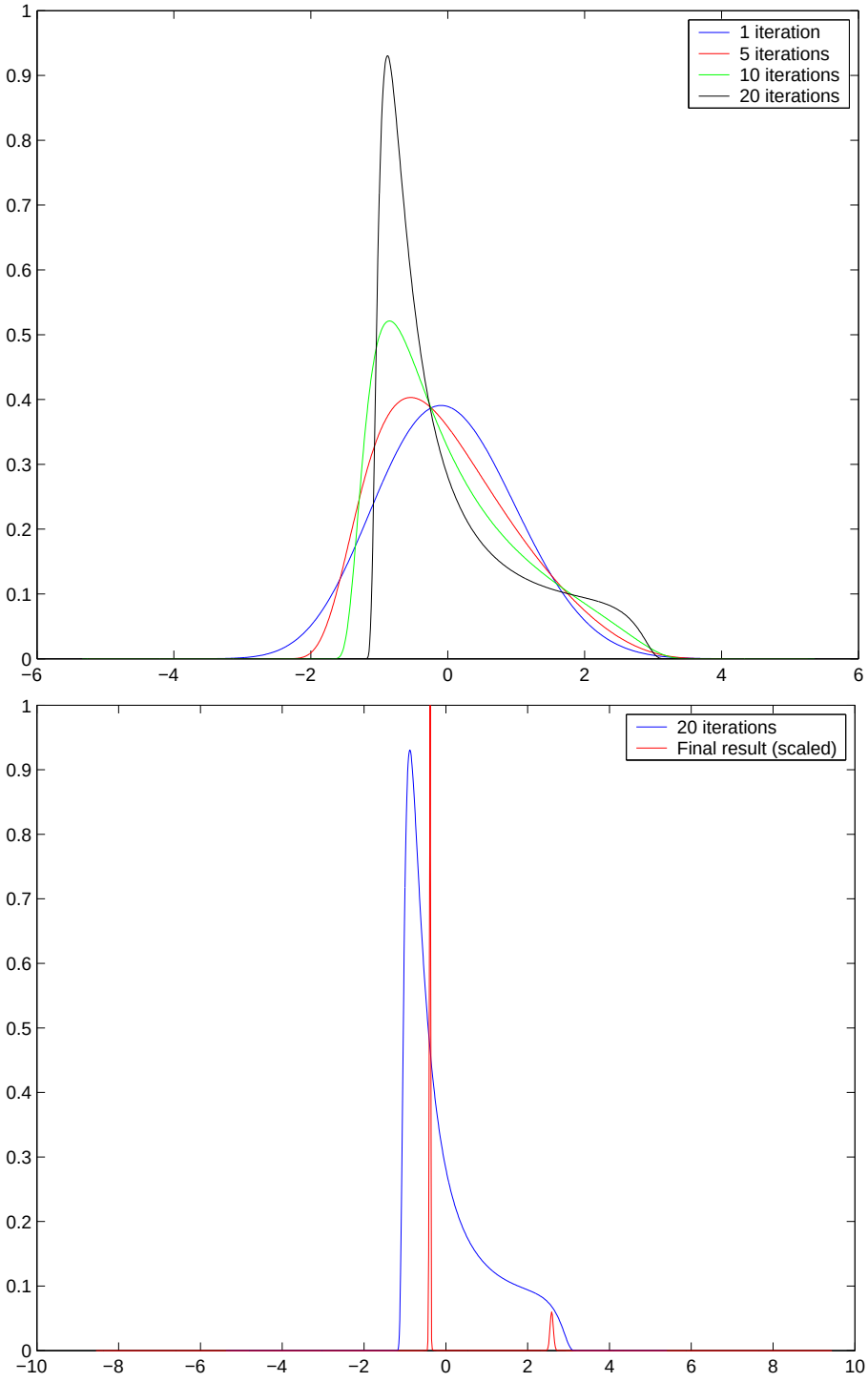


Figure 6.3: *Iterations of an empirical method for obtaining the distribution giving the worst tail error*

$P(X_1 + X_2 > 2c) = 0$ which is obviously not the maximum. There are now two other possibilities.

Case 1: $a - 1/a < 2c$, in which case a transformation which replaces $\delta(x-a)$ by $\delta(x-c)$ can be made which reduces the mean and variance, while keeping $P(X_1 + X_2 \geq 2c)$ constant. The distribution for which this is a maximum must therefore be invariant to this transformation, so $a = c$.

Case 2: $a - 1/a \geq 2c$, where again a transformation replacing $\delta(x-a)$ by $\delta(x-2c+1/a)$ can be made which reduces the mean and variance, while keeping $P(X_1 + X_2 \geq 2c)$ constant. The distribution for which this is a maximum must therefore be invariant to this transformation, so $a - 1/a = 2c$ which implies $a = c + \sqrt{1+c^2}$.

Numerically, it has been found that Case 1 gives the better solution for $c < 0.9061$ but that Case 2 is optimal in the remaining cases. For prescreeners, the value of c will usually be set quite high, so Case 2 is of most interest.

There are many intuitive reasons why Hypothesis 1 should be considered valid, which indicates that there should be some simple argument allowing it to be proved rigorously. The only proof found to date is somewhat cumbersome, and relies on the following theorem.

Theorem 2: Suppose $f(x)$ is the distribution function for i.i.d. variables X_i such that $P(X_1+X_2 > 2c)$ is a maximum. Then we can decompose f into two non-negative functions $f(x) = r(x) + s(x)$ such that $r(x) = 0 \; \forall x > c$ and $s(x)$ is symmetrical about $x = c$.

Proof: Assume that $f(x+c) > f(c-x)$ for $x \in (\xi_1, \xi_2]$. Now we consider a transformation which replaces $f(x)$ on the interval $[c-\xi_2, c-\xi)$ by a δ-function with identical area at $x = c-\xi$. To make the transformation symmetric about $x = c$ (as required by Theorem 1), $f(x)$ on the interval $(c+\xi, c+\xi_2]$ should similarly be replaced by a δ-function at $x = c+\xi$. The decrease in the mean due to this transformation will be

$$\begin{aligned} h(\xi) &= \int_{\xi}^{\xi_2} (x+c) f(x+c) dx + \int_{\xi}^{\xi_2} (c-x) f(c-x) dx \\ &\quad -(c+\xi) \int_{\xi}^{\xi_2} f(x+c) dx - (c-\xi) \int_{\xi}^{\xi_2} f(c-\xi) dx \\ &= \int_{\xi}^{\xi_2} (x-\xi)(f(x+c) - f(c-x)) dx \end{aligned}$$

which is greater than zero for $\xi = \xi_1$. In fact, because the function is continuous, there will exist some $\xi < \xi_1$ for which $h(\xi) > 0$. Because the transformation satisfies the conditions of Theorem 1, $P(X_1 + X_2 < 2c)$ is unchanged, while the mean (and obviously the variance too) is decreased. Thus, after renormalisation, the new distribution function will have a larger $P(X_1 + X_2 < 2c)$. This leads to a contradiction, unless the original assumption is wrong, and therefore $f(x+c) \leq f(c-x) \; \forall x$ and the functions $r(x)$ and $s(x)$ defined in the theorem must exist.

One consequence of the above theorem is that a value of ξ in equation (10) can always be found for distribution functions of interest such that $\mu = 0$ in equation (8). This is because from Theorem 2,

$$\begin{aligned}
\mu &\approx -\varepsilon \int_{-\infty}^{\infty} f(x+c)h(x+c)dx \\
&= -\varepsilon \int_{-\infty}^{\infty} (r(x+c)+s(x+c))h(x+c)dx \\
&= -\varepsilon \int_{-\infty}^{c} r(x+c)h(x+c)dx \\
&= \kappa \overset{lim}{\rightarrow} \infty \int_{-\kappa}^{c} r(x)h(x)dx.
\end{aligned}$$

Now when $x < c$, $h(x) > 0$ for $x \in (c-\epsilon, c)$ and $h(x) \leq 0$ otherwise. This means that by setting $\xi = 0$, μ will be negative and by setting $\xi > \kappa$, that μ will be positive. Since the function is continuous, then there must be some intermediate value for ξ for which $\mu = 0$. For the optimal distribution however, the transformation $g(x) = x + \varepsilon h(x)$ must not affect the variance, so from equation (9),

$$\int_{-\infty}^{\infty} x f(x) h(x) dx = 0. \tag{11}$$

One possibility is that $f(x)$ consists only of δ functions at the zero crossings of $h(x)$, so that both the mean and variance of the function are unaffected by the perturbation. This possibility is consistent with Hypothesis 1. Since the above argument is not specific to the form of $h(x)$ in equation (10), but can be applied more generally to any function that is odd about $x = c$ with an odd number of zero crossings for $x < c$. Equation (11) must hold true for all of these functions $h(x)$, so it is intuitively unlikely that there is another function which satisfies the required optimality conditions. The only conclusive proof found so far is somewhat more cumbersome than the preceding argument, and proceeds as follows:

Proof of Hypothesis 1: Suppose the distribution f is decomposed as a mixture of two other distributions g and h such that $g(x) = 0 \ \forall x < c$ and $h(x) = 0 \ \forall x \geq c$. Define the proportion of g to be G and to have mean and variance μ_g, σ_g^2, while h has mean and variance μ_h, σ_h^2. Now due to the constraints on the mean and variance of the overall distribution, we have

$$G\mu_g + (1-G)\mu_h = 0 \ \Rightarrow \mu_h = \frac{-G}{1-G}\mu_g$$

for the mean and

$$G(\mu_g^2 + \sigma_g^2) + (1-G)(\mu_h^2 + \sigma_h^2) = 1 \ \Rightarrow \sigma_h^2 = \frac{1 - G(\mu_g^2 + \sigma_g^2)}{1-G} - \mu_h^2$$

for the variance. Now from the one-tailed Chebychev inequality,

$$\begin{aligned} P(X+X \geq 2c) &= G^2 P(X_g + X_g > 2c) + 2G(1-G)P(X_g + X_h > 2c) \\ &\leq G^2 + 2G(1-G)\frac{\sigma_g^2 + \sigma_h^2}{\sigma_g^2 + \sigma_h^2 + (2c - \mu_g - \mu_h)^2} \\ &= \hat{P}. \end{aligned}$$

The problem now is to find distributions which maximise the upper bound $\hat{P}$, and then show that this bound is also a strict bound on $P(X + X \geq 2c)$. This is a constrained optimisation problem, where the only important constraints are $\sigma_g^2 \geq 0$, $\sigma_h^2 \geq 0$ and $\mu_g \geq c$. Other constraints (such as $G \leq 1$) are enforced by the previous constraints, so need not be considered. If the optimal solution is inside the feasible region, then the upper bound will be a maximum when $\partial/\partial\sigma_g^2\left(\hat{P}\right) = 0$, which according to MAPLE gives either

- **Case 1:** $G = 0$, for which the upper bound is zero. This is obviously a minimum rather than a maximum.
- **Case 2:** $G = 1/2$, for which

  $$\hat{P} = \frac{3}{4} + \frac{c^2}{\mu_g^2 - 1 - 2c^2}$$

  so that $\hat{P} \to \infty$ as $\mu_g \to \sqrt{1 + 2c^2}$. Over the feasible region, $\hat{P} \leq 1$, so this maximum cannot be inside the feasible region.
- **Case 3:** $\mu_g = (2c(1-G))/(1-2G)$, for which $\hat{P} = (2-G)G$ which is a maximum for $G = 1$. This is not feasible because $G \leq 1/(1 + c^2)$ from the one-tailed Chebychev inequality.

Because $\hat{P}$ doesn't achieve a maximum strictly inside the feasible region, then the maximum feasible solution must lie on one of the constraint surfaces, which leaves three subcases to consider:

- **Case 4a:** If $\sigma_h^2 = 0$ is the constraint, then the variance constraint will fix σ_g as a function of G and μ_g, which makes

  $$\hat{P} = G^2 + \frac{2G(1-g)^2(1 - G - G\mu_g^2)}{(1-G)^2(1 + 4c^2G) - 4c(1-G)G(1-2G)\mu_g + G^2(4G-3)\mu_g^2}.$$

  Now this is a maximum when

  $$\begin{aligned} \frac{\partial \hat{P}}{\partial \mu_g} &= \frac{4(1-G)^3G^2\left(\mu_g(1-2G) - 2c(1-G)\right)\left(2G(1 + c\mu_g) - 1\right)}{\left((1-G)^2(1 + 4c^2G) - 4c(1-G)G(1-2G)\mu_g + G^2(4G-3)\mu_g^2\right)^2} \\ &= 0. \end{aligned}$$

  This has two solutions

- **Case 4aa:** $\mu_g = 2c(1-G)/(1-2G)$, which means $\hat{P} = G(2-G)$ which achieves a maximum at $G = 1$ which is outside the feasible region. This means that one of the other constraints must also be satisfied, so $\sigma_g = 0$ or $\mu_g = c$ (which also implies $\sigma_g = 0$). This maximum will therefore occur for a distribution consisting of two δ-functions, as in the hypothesis.
- **Case 4ab:** $\mu_g = (1-2G)/(2cG)$, which means that

$$\hat{P} = \frac{4(1+c)^2G^3 - (5+4c^2)G^2 + 4G - 2}{4G - 4(1-G)c^2 - 3}.$$

which has its optimum where

$$\frac{\partial \hat{P}}{\partial G} = \frac{32(1+c^2)^2G^3 + 8(1+c^2)(7-c^2)G^2 + (32(1+c^2)^2 - 2)G - 8(1+c^2) + 4}{(4G - 4(1-G)c^2 - 3)^2} = 0.$$

The numerator, being a cubic, has potentially three zeros. A maximum will occur when $\partial\hat{P}/\partial G$ crosses from positive to negative. Since the coefficient of G^3 is positive, the only maximum would correspond to the middle zero. If such a zero exists, it must occur between the local minimum and local maximum of the cubic. These occur at

$$G = \frac{4c^2+5}{12(1+c^2)} \text{ and } G = \frac{12c^2+9}{12(1+c^2)},$$

but the numerator is positive at both of these points. This means that the cubic has only one real zero, which corresponds to a minimum of $\hat{P}$. The global maximum, occurring as $G \to \infty$, is outside of the feasible region so, as in Case 4aa, the feasible maximum will occur when another constraint ($\sigma_g = 0$) is satisfied, and the distribution becomes a mixture of two δ-functions.

- **Case 4b:** If $\sigma_g^2 = 0$, then

$$\hat{P} = \frac{G\left(4G^2(c-\mu_g^2) + G(1-4c^2+4c\mu_g+\mu_g^2) - 2\right)}{(4c(1-2G)\mu_g - 4c^2(1-G) - (1-4G)\mu_g^2 - 1}$$

After differentiating with respect to μ_g, the numerator will be

$$4G\left((1-2G)\mu_g - 2c(1-G)\right)\left(2G(1+c\mu) - 1\right).$$

which will equal zero at the maximum, and so there are three possibilities

- **Case 4ba:** $G = 0$ which, as mentioned previously, is a minimum rather than a maximum.
- **Case 4bb:** $G = (2c-\mu_g)/(2(c-\mu_g))$, where

$$\hat{P} = 1 - \frac{\mu_g^2}{4(c-\mu_g)^2},$$

which increases monotonically with μ_g, so the maximum would occur as $\mu_g \to \infty$ (where $G \to 1$), which is outside the feasible region. This means that the maximum over the feasible region must also lie on another constraint boundary.

– **Case 4bc:** $G = 1/(2(1 + c\mu_g))$, where

$$\hat{P} = \frac{3 + c(2c + \mu_g)}{4(1 + c\mu_g)^2(1 + 2c^2 - c\mu_g)}.$$

The maximum occurs when $\partial\hat{P}/\partial\mu_g = 0$, which will be true when

$$\mu_g = \frac{-2 - c^2 \pm \sqrt{5}(1 + c^2)}{c}.$$

Only the '+' solution is feasible, for which

$$\hat{P} = \frac{11 + 5\sqrt{5}}{32}\frac{1}{(1 + c^2)^2}.$$

While this solution may be a maximum, it is less than $1/(1 + c^2)^2$ which is an achievable value for $P(X + X > 2c)$ based on two δ-functions. If it is a maximum, it is only local.

- **Case 4c:** When $\mu_g = c$, $\hat{P}$ will be a maximum when the distribution h has the largest possible value for $P(X_h > c)$. This corresponds to the solution

$$f(x) = A\delta(x - c) + (1 - A)\delta(x + \frac{1}{c}).$$

In all of the above feasible cases, the optimum distribution function can be written as a sum of two delta functions, as described in Hypothesis 1.

6.2.4.2 The N point case

If x_i is a set of pixel values which have had their means removed, then when N is even, the signal mean $Z_N = \sum_{i=1}^{N} x_i/N$ can be written as a sum of two random variables with the same statistics as $Z_{N/2}/2$. This means that Hypothesis 1 may be used to estimate the upper limit on $P(Z_N > k\sigma)$ for some fixed threshold k. As with equation (7), the variance of $Z_{N/2}/2$ will be $\sigma^2/(2N)$ and so the fraction of false alarms will be

$$P(Z_N > k\sigma) = P\left(Z_{N/2}/2 + Z_{N/2}/2 > 2\left(k\sqrt{\frac{N}{2}}\right)\frac{\sigma}{\sqrt{2N}}\right)$$

which from Hypothesis 1 (and for large c), can be written as

$$P(Z_N > k\sigma) = \frac{1 + 2a^2}{(1 + a^2)^2}$$

where

$$a = \sqrt{1 + \frac{N}{2}k^2} + k\sqrt{\frac{N}{2}}.$$

As with the simplified assumptions in Subsection 6.2.3, if the number of pixels in the target can be improved by a factor of N by improving the resolution, the number of pixels in the image also increases by a factor of N, so the false alarm rate ($/km^2$) will be $NP(Z_N > k\sigma)$. Again, in the limit as $N \to \infty$, $a \to \sqrt{2N}k$ and so

$$\lim_{N \to \infty} NP(Z_N > k\sigma) = \frac{2Na^2}{a^4} = 1/k^2$$

which means that the false alarm rate is not guaranteed to decrease as the resolution of the image improves.

The form of the optimal solution for two points, as described by Hypothesis 1, suggests that a similar form of solution may also be optimal for the N point scenario. This leads to the following purely speculative hypothesis

Hypothesis 2: If $f(x)$ is the distribution function of i.i.d. random variables X_i (each having zero mean and unit variance) which gives the largest value for $P(\sum_i X_i > Nc)$, then it will be of the form

$$f(x) = \frac{1}{1+a^2}\delta(x-a) + \frac{a^2}{1+a^2}\delta(x+1/a)$$

where either $a = c$, or $a = (Nc + \sqrt{N^2c^2 + 4(N-1)})/2$ (obtained by assuming $a - (N-1)/a = Nc$).

The above hypothesis is guaranteed to at least give a lower bound to the upper bound, which can still prove of use. Now, as $N \to \infty$, the positions of the rightmost delta function spike will be either at $a = c$ or $a \approx Nc(1 + \sqrt{1 + 4/(Nc^2)})/2$. The corresponding magnitudes of each spike will be $1/(1 + a^2)$, which results in

- **Case 1:** When $a = c$, the sum $\sum_i X_i$ will only exceed Nc when every independent variable $X_i = c$, so

  $$P(\sum_i X_i \geq Nc) = \frac{1}{(1+c^2)^N}$$

  Choosing $c = k/\sqrt{N}$, and looking at the limit as $N \to \infty$ gives

  $$P(Z > k) = N \overset{\lim}{\to} \infty \frac{1}{(1 + k^2/N)^N} \approx N \overset{lim}{\to} \infty (1 - \frac{k^2}{N})^N = \exp(-k^2).$$

  where Z is the average of the random variables, normalised to have zero mean and unit variance.

- **Case 2:** In the remaining case, when $a \approx Nc(1+\sqrt{1+4/(Nc^2)})/2$, the sum $\sum_i X_i$ will not exceed Nc only when each independent variable has $X_i = -1/a$, which means

$$P(\sum_i X_i \geq Nc) = 1 - \left(1 - \frac{1}{(1+a^2)}\right)^N.$$

Again, choosing $c = k/\sqrt{N}$, and looking at the limit as $N \to \infty$ gives

$$\begin{aligned} P(Z > k) &= \overset{\lim}{N \to \infty}\, 1 - \left(1 - \frac{1}{1 + Nk^2\left(1+\sqrt{1+4/k^2}\right)^2/4}\right)^N \\ &= \overset{\lim}{N \to \infty}\, 1 - \left(1 - \frac{4}{N\left(k+\sqrt{k^2+4}\right)^2}\right)^N \\ &= 1 - \exp\left(-\frac{4}{\left(k+\sqrt{k^2+4}\right)^2}\right) \end{aligned}$$

Figure 6.4 shows plots of the hypothesised bounds on the tail probabilities as the number of independent components N is increased. The outer two blue plots (for $N = 1$ and $N = 2$) are the exact upper bounds, while the remaining blue curves are lower bounds for the upper bound for $N = 2^i, i = 2 \ldots 10$. The two limiting cases for large N, as described above, are also plotted. It is interesting to note that as $N \to \infty$, the probability bound does not actually tend towards that for a Gaussian, as might be expected from the central limit theorem. This is because for any given value of N, a distribution function $f(x)$ can be found so that the sum differs significantly from a normal distribution. As a result,

$$\overset{\max}{f(x)} \overset{\lim}{N\to\infty} P\left(\sum_{i=1}^N X_i > k | X_i \sim f(x)\right) \neq \overset{\lim}{N\to\infty} \overset{\max}{f(x)} P\left(\sum_{i=1}^N X_i > k | X_i \sim f(x)\right).$$

The left hand side will correspond to the expected result from the central limit theorem, while the right hand side is the result plotted in Figure 6.4.

As with the one and two independent component cases, Hypothesis 2 can provide an approximate expression for the variation of the false alarm rate of a prescreener with the image resolution. The worst case false alarm rate will be proportional to

$$NP(Z > k\sqrt{N}) = N\left(1 - \exp\left(-\frac{4}{\left(k\sqrt{N}+\sqrt{Nk^2+4}\right)^2}\right)\right),$$

which as $N \to \infty$ becomes

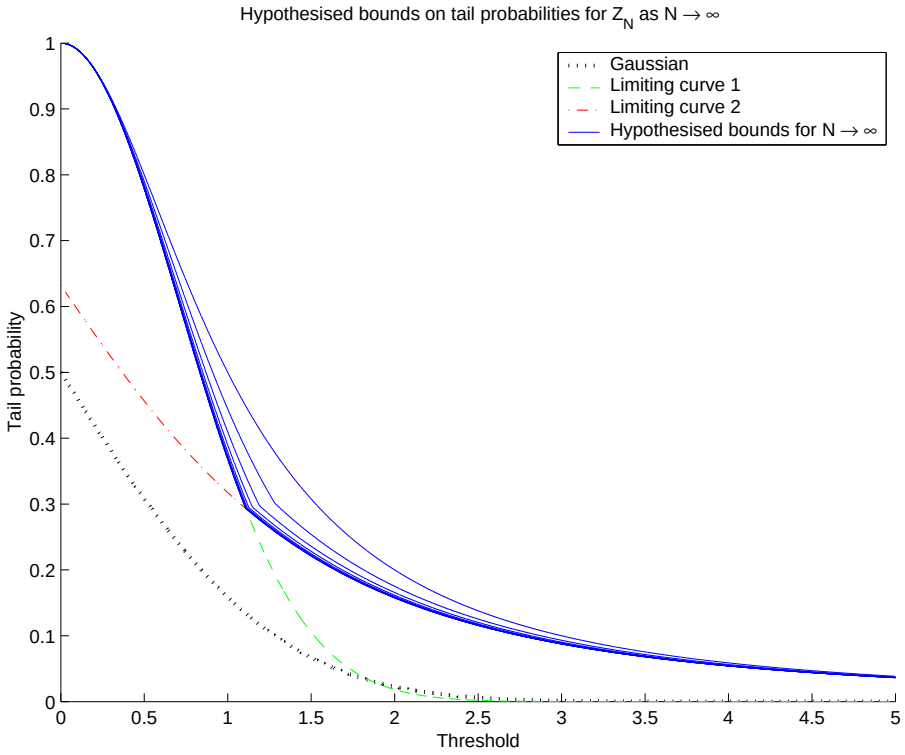


Figure 6.4: *Variation of tail probabilities with number of independent components*

$$N\left(1-\exp\left(-\frac{1}{k^2N}\right)\right)=\frac{1}{k^2}+\mathcal{O}\left(\frac{1}{N}\right),$$

which is the same as for the two point case. This means that using knowledge of the mean and covariance alone, it can only be shown that the false alarm rate of a prescreener will not increase (at least in the limit) as the resolution of the imagery is improved.

The above result is based on the worst possible distribution function. While similar bounding techniques have been used to measure performance of a wide variety of techniques (such as for measuring computational performance of algorithms for solving various combinatoric or sorting problems), there has been a shift in emphasis more recently towards average rather than worst case performance. Similar considerations for the current prescreening problem may result in a lower and more useful bound on the false alarm rate, although in this context it is far from clear how an "average distribution" might be defined. On the other hand, the above arguments assumed that the variance of the background distribution remained unchanged as the resolution was improved. In practice, the amount of speckle (and hence the variance) tends to increase with improved resolution, which would tend to increase the observed false alarm rate.

6.2.5 Template prescreening

The prescreeners described in the previous subsections assumed independent background pixels. When a single pixel signal window is used, the extent of the target is not taken into account, thus limiting the potential of the prescreener to detect targets. The prescreeners described so far which use multiple pixels in the signal window all compare the unweighted mean of the pixels with the background. Such a prescreener is effectively performing a one-tailed hypothesis test against the background distribution. Effective classification however requires that the distribution of target pixels also be considered. This means that each of the pixels within the signal window will be given a weighting. When the weighting of all pixels is zero except for the middle pixel, this results in the standard single pixel Gaussian CFAR detector. When all the weights are equal, the mean value will be used.

The ADSS currently contains a "linear-discriminant" prescreening module similar to the above idea. In this module, a set of training data comprised of rectangular images containing either known targets or probable background clutter is used to train a $\beta-$ linear discriminant [5] to determine a set of weights. Then when using the weights in a prescreener, the 2D weight function is convolved with the image prior to applying a global threshold.

Suppose a training set of $M \times M$ images is available, and there are only N_t target examples and N_b background examples in the training set (to make a total of $N = N_t + N_b$ images). The ADSS currently replaces each $M \times M$ image by a vector of length M^2, and then calculates mean μ and the $M^2 \times M^2$ covariance matrix Σ for both the target and background classes. Applying the $\beta-$linear discriminant to the two classes (which involves computing a matrix inverse of the form $(\Sigma_b + \beta\Sigma_t)^{-1}$), yields a weight vector $\mathbf{w}$ corresponding to the required 2D weight function over an $M \times M$ rectangular support. A global threshold is then used to detect the targets.

The approach taken in this subsection differs from the current ADSS approach in a number of ways. Firstly, it is adaptive so that the detection performance will be invariant to linear changes in intensity scale. Secondly, it may be used to separate an arbitrary number of classes, which allows it to be used in low level classification as well as in prescreening. Thirdly, it is more robust when only small amounts of data are available. For the maritime detection problem, the number of images containing known targets is relatively low (meaning that Σ_t is likely to be singular), there is generally no shortage of background so the sum $\Sigma_b + \beta\Sigma_t$ should stay non-singular. The difficulty will arise when considering large window sizes M, which will be necessary when considering higher resolution imagery. In this case, either the number of pixels to consider may easily be larger than the number of training images available and the covariance matrices become singular, or the computational cost for inverting the matrix becomes prohibitive. The following subsection describes an algorithm for finding the required linear discriminant in a computationally efficient manner, while at the same time artificially inflating the number of training examples considered by the addition of white noise.

6.2.5.1 Calculation of template weights

Suppose there are N images, each containing M^2 pixels where $N < M^2$ so that the covariance matrix will be singular. The corresponding $M \times M$ training vectors will be defined by $\mathbf{X}_i$ for $i = 1 \ldots N$. Each training vector is associated with a class $C_i \in \{1 \ldots C_{max}\}$ (where $C_{max} = 2$ for the detection problem, which separates targets from background clutter).

Due to the sparsity of data, no information is available in the pixel space outside of the N dimensional hyperplane containing all of the data points. Thus no information is lost by reducing the dimensionality to the N containing the data, giving a new set of training vectors $\mathbf{Y}_i$ whose jth component is defined by

$$Y_{i,j} = (\mathbf{X}_i)^T \mathbf{X}_j.$$

A linear discriminant $\mathbf{W}^T\mathbf{Y} = c$ can now be found in this reduced space more easily since the covariances will no longer be singular and will only be of size $N \times N$. Expressing the discriminant in terms of the original training vectors gives

$$\begin{aligned}
\mathbf{W}^T\mathbf{Y} &= \sum_j W_j Y_{,j} \\
&= \sum_j W_j \sum_k X_{,k} X_{j,k} \\
&= \sum_k \left(\sum_j W_j X_{j,k} \right) X_{,k} \\
&= \sum_k w_k X_{,k}
\end{aligned}$$

So the weight vector for the reduced space can then be transformed to the original pixel space using $w_i = \sum_j W_j X_{i,j}$, and the problem has been changed from M^2 dimensional to N dimensional. Because generalisation errors of most classifiers increases with dimensionality, it is desirable to further reduce the dimensionality even further. One commonly used technique is PCA (Principal Component Analysis) which captures the dimensions which produce the maximum variability in the data by choosing the eigenvectors of the covariance matrix with the largest eigenvalues. From a classification point of view however, the directions of greatest class separation is much more useful than directions of greatest variability in the data, so PCA often produces poor results. An alternative method is used by Sato [16] who reduces the dimensionality to N_f features by defining an $N_f \times N$ dimension reduction matrix, and providing a measure of the classifier error in the low dimensional space

$$\text{Error} = \inf_{\theta} f(\mathbf{AY}, \theta).$$

where the vector θ correspond to the parameters of the discriminant. By minimising this expression with respect to $\mathbf{A}$, a more useful dimension reduction can be achieved from

a classification point of view. The error function given in Sato's paper was a smoothed measurement of the classification error of a Learning Vector Quantisation (LVQ) classifier, where the smoothing parameter was gradually changed so that the actual training error was minimised by the dimension reduction. This formulation is somewhat complicated, and it is difficult to incorporate robustness to white noise (as described in the next subsubsection). In this report, a simpler error function will be used.

Suppose that in the N dimensional space that the class i has mean μ_i and covariance Σ_i. Since discrimination will eventually need to be done in the original M^2 dimensional space, a classifier which can be easily transformed between spaces will need to be used. For this reason, a simple pairwise linear discriminant is used here. While Fisher's linear discriminant could have been used, this involves inversion of covariance matrices of size N which may still be quite large, and so was avoided. Instead, a naive discriminant was used, which separated classes i and j by a hyperplane having normal $\mu_i - \mu_j$. The bias was chosen by assigning weights w_i to class i and then choosing the hyperplane separating classes i and j to go through the point $(w_i\mu_i + w_j\mu_j)/(w_i + w_j)$. Now the variance of class i in the direction $\mu_j - \mu_i$ will be

$$\begin{aligned}
\sigma_i^2 &= E\left(\left(\frac{(\mu_i-\mu_j)^T Y}{|\mu_i-\mu_j|}\right) - E\left(\frac{(\mu_i-\mu_j)^T Y}{|\mu_i-\mu_j|}\right)\right)^2 \\
&= \frac{1}{|\mu_i-\mu_j|^2} E\left((\mu_i-\mu_j)^T(Y-\mu_i)\right)^2 \\
&= \frac{1}{|\mu_i-\mu_j|^2}(\mu_i-\mu_j)^T E\left((Y-\mu_i)^T(Y-\mu_i)\right)(\mu_i-\mu_j) \\
&= \frac{(\mu_i-\mu_j)^T\Sigma_i(\mu_i-\mu_j)}{|\mu_i-\mu_j|^2}
\end{aligned}$$

Also the perpendicular distance from μ_i to the hyperplane will be

$$\begin{aligned}
c_i &= \frac{1}{|\mu_i-\mu_j|}(\mu_i-\mu_j)^T\left(\mu_i - \frac{w_i\mu_i + w_j\mu_j}{w_i+w_j}\right) \\
&= \frac{w_i}{w_i+w_j}|\mu_i-\mu_j|
\end{aligned}$$

Now if the class means and covariances have been accurately measured, then the upper bound on the classification error for class i due to this linear discriminant can be determined from the formula in Cooke and Peake [4] to be

$$\begin{aligned}
\text{Error bound} &= \frac{\sigma_i^2}{\sigma_i^2 + c_i^2} \\
&= \frac{(\mu_i-\mu_j)^T\Sigma_i(\mu_i-\mu_j)}{(\mu_i-\mu_j)^T\Sigma_i(\mu_i-\mu_j) + \left(\frac{w_i}{w_i+w_j}\right)^2((\mu_i-\mu_j)^T(\mu_i-\mu_j))^2}
\end{aligned}$$

Now if a dimension reduction matrix $\mathbf{A}$ is applied to the training vectors before the linear discriminant is applied, then the means and covariances for class i become $\mathbf{A}^T\mu_i$ and $\mathbf{A}^T\Sigma_i\mathbf{A}$, and so the total error over all classes due to all linear discriminants must be less than

$$\sum_i \sum_{j\neq i} \frac{(\mu_i-\mu_j)^T\mathbf{A}\mathbf{A}^T\Sigma_i\mathbf{A}\mathbf{A}^T(\mu_i-\mu_j)}{(\mu_i-\mu_j)^T\mathbf{A}\mathbf{A}^T\Sigma_i\mathbf{A}\mathbf{A}^T(\mu_i-\mu_j)+\left(\frac{w_i}{w_i+w_j}\right)^2\left((\mu_i-\mu_j)^T\mathbf{A}\mathbf{A}^T(\mu_i-\mu_j)\right)^2}$$

which should be minimised with respect to the discriminant parameters w_i as well as the feature reduction matrix $\mathbf{A}$. For the current case when the number of training vectors Y is equal to the dimensionality of the vectors, it is always possible to choose a matrix $\mathbf{A}$ such that the matrix upper bound is zero, and the linear discriminants will overfit the data. In fact, an infinite number of such matrices $\mathbf{A}$ exist. One solution can be found analytically by solving the set of linear equations $\mathbf{Y}_i A = C_i$ for $i = 1 \ldots N$ where C_i is the class label for the training vector $\mathbf{Y_i}$. An alternative is to use a gradient descent type method. Obtaining the gradient for $\mathbf{A}$ involves the following differentiation

$$\begin{aligned}\frac{\partial}{\partial A_{ij}}\mu^{\mathbf{T}}\mathbf{A}\mathbf{A}^{\mathbf{T}}\mathbf{\Sigma}\mathbf{A}\mathbf{A}^{\mathbf{T}}\mu &= \frac{\partial}{\partial A_{ij}}\mu_k A_{kl}A_{ml}\Sigma_{mn}A_{np}A_{qp}\mu_q \\ &= \mu_i A_{mj}\Sigma_{mn}A_{np}A_{qp}\mu_q + \mu_k A_{kj}\Sigma_{in}A_{np}A_{qp}\mu_q \\ &\quad +\mu_k A_{kl}A_{ml}\Sigma_{mi}A_{qj}\mu_q + \mu_k A_{kl}A_{ml}\Sigma_{mn}A_{nj}\mu_i.\end{aligned}$$

where repeated subscript tensor notation has been used. By relabelling the dummy variables, and using the fact that the covariance matrix $\mathbf{\Sigma}$ will be symmetric, the first and last terms can be shown to be equivalent. Similarly, the second and third terms are equivalent, so

$$\begin{aligned}\frac{\partial}{\partial A_{ij}}\mu^{\mathbf{T}}\mathbf{A}\mathbf{A}^{\mathbf{T}}\mathbf{\Sigma}\mathbf{A}\mathbf{A}^{\mathbf{T}}\mu &= 2\left(\mu_i A_{mj}\Sigma_{nm}A_{np}A_{qp}\mu_q + \mu_k A_{kj}\Sigma_{in}A_{np}A_{qp}\mu_q\right) \\ &= 2\left(\mu_i(\mu^{\mathbf{T}}\mathbf{A}\mathbf{A}^{\mathbf{T}}\mathbf{\Sigma}\mathbf{A})_j + (\mathbf{\Sigma}\mathbf{A}\mathbf{A}^{\mathbf{T}}\mu\mu^{\mathbf{T}}\mathbf{A})_{ij}\right)\end{aligned}$$

By a similar argument, it can be shown that

$$\frac{\partial}{\partial \mathbf{A}}\mu^{\mathbf{T}}\mathbf{A}\mathbf{A}^{\mathbf{T}}\mu = 2\mu\mu^{\mathbf{T}}\mathbf{A}.$$

This means that the new point after an iteration of gradient descent will be $\mathbf{A} \rightarrow \mathbf{A} + \epsilon\Delta\mathbf{A}, \mathbf{W} \rightarrow \mathbf{W} + \epsilon\Delta\mathbf{W}$ where ϵ is the step size and

$$\begin{aligned}\Delta\mathbf{A} &= 2\sum_i\sum_{j\neq i}\frac{(\mathbf{M_{ij}}+\mathbf{M_{ij}}^T)\mathbf{A}}{K_{ij}+L_{ij}} - \frac{K_{ij}\left((\mathbf{M}_{ij}+\mathbf{M}_{ij}^T)\mathbf{A}+\mathbf{N}_{ij}\right)}{(K_{ij}+L_{ij})^2} \\ \Delta W_k &= 2\sum_{j\neq k}\left(\frac{K_{jk}L_{jk}}{(K_{jk}+L_{jk})^2(W_j+W_k)} - \frac{2K_{kj}L_{kj}W_k}{(K_{kj}+L_{kj})^2W_j(W_j+W_k)}\right)\end{aligned}$$

where

$$\begin{aligned}
K_{ij} &= (\mu_i - \mu_j)^T \mathbf{A}\mathbf{A}^\mathbf{T}\mathbf{\Sigma_i}\mathbf{A}\mathbf{A}^\mathbf{T}(\mu_i - \mu_j) \\
L_{ij} &= \left(\frac{W_i}{W_i + W_j}\right)^2 \left((\mu_i - \mu_j)^T \mathbf{A}\mathbf{A}^\mathbf{T}(\mu_i - \mu_j)\right)^2 \\
\mathbf{M}_{ij} &= (\mu_i - \mu_j)(\mu_i - \mu_j)^T \mathbf{A}\mathbf{A}^\mathbf{T}\mathbf{\Sigma}_i \\
\mathbf{N}_{ij} &= \left(\frac{W_i}{W_i + W_j}\right)^2 (\mu_i - \mu_j)(\mu_i - \mu_j)^T \mathbf{A}
\end{aligned}$$

6.2.5.2 Robustness to white noise

Recognition of patterns within images is based on correlations between groups of neighbouring pixels. The addition of small amounts of uncorrelated noise tends to have relatively little effect on the ability of humans to identify objects, and so the addition of white noise may be useful in artificially increasing the size of training set sizes. For an $M \times M$ image however, there need to be at least M^2 training vectors before difficulties with the sparseness of the data set can be overcome. Finding good discriminants with this many data points however often becomes computationally prohibitive. Fortunately, it is possible to adapt the algorithm from the previous subsubsection to use an effectively infinite size training set generated by the addition of white noise, without the necessity of storing all possible instances.

Consider a set of images (given by vectors $\mathbf{X}_i$) corrupted by white noise with variance σ^2, to give $j = 1 \ldots J$ noisy images $\hat{\mathbf{X}}_{i,j} = \mathbf{X}_i + \sigma\mathbf{n}_j$. Again, suppose that the image domain is reduced to the subspace containing the known data, so that the kth component in the reduced space is

$$\hat{Y}_{i,j,k} = (\mathbf{X}_k)^T \hat{\mathbf{X}}_{i,j} = (\mathbf{X}_k)^T (\mathbf{X}_i + \sigma\mathbf{n}_j) = Y_{i,k} + \sigma(\mathbf{X}_k)^T \mathbf{n}_j.$$

The mean value of $\hat{\mathbf{Y}}_{i,j}$ (μ_Y) will be the same as that obtained previously. The covariance however will be

$$\begin{aligned}
\hat{\Sigma}_{k,l} &= E_i(E_j((\hat{Y}_{i,j,k} - \mu_{Y_k})(\hat{Y}_{i,j,l} - \mu_{Y_l}))) \\
&= E_i\left(E_j\left((Y_{i,k} - \mu_{Y_k} + \sigma\mathbf{X}_k^T\mathbf{n}_j)(Y_{i,l} - \mu_{Y_l} + \sigma\mathbf{X}_l^T\mathbf{n}_j)\right)\right) \\
&= E_i((Y_{i,k} - \mu_{Y_k})(Y_{i,l} - \mu_{Y_l})) \\
&\quad +\sigma\left(E_i((Y_{i,k} - \mu_{Y_k})\mathbf{X}_l^T)E_j(\mathbf{n}_j) + E_i((Y_{i,l} - \mu_{Y_l})\mathbf{X}_k^T)E_j(\mathbf{n}_j)\right) \\
&\quad +\sigma^2\mathbf{X}_k E_j(n_j n_j^T)\mathbf{X}_l^T \\
&= \Sigma_{k,l} + \sigma^2\mathbf{X}_k\mathbf{X}_l^T
\end{aligned}$$

Using these new means and covariances in the algorithm discussed in the previous subsubsection should prevent the error bound of zero from being reached so that overfitting

is reduced. One down side to this method is that the correct value for σ is not able to be chosen *a-priori*, and it is now required to choose a value.

6.2.5.3 Classification results

To test the performance of the noise robust template matcher, it was applied to two different sets of data. The first set was the Lincoln Lab's MSTAR dataset, and the second was a group of spotlight SAR images of ships from INGARA. Although the MSTAR data set is focussed on land targets instead of maritime targets, which is the focus of this report, results for this data are presented here for two reasons. Firstly the INGARA data is quite noisy due to the presence of Doppler smearing, and is not well centered. Since it is useful to know that a method works in the best possible case with well centered targets and high signal to noise ratio, the INGARA data is not suitable for this purpose. Secondly, it is useful to compare classifier results with others in the literature, and the MSTAR database has been used considerably in the literature which allows such comparison more easily.

The MSTAR database contains images of three types of land vehicles, the T72 tank, and the BTR70 and BMP2 armoured personnel carriers. The images are split into a training set of 1622 images taken at a 17^o look-down angle, while the test set contains 1365 images from a 15^o look-down angle. Each image is 128×128 pixels and has 1ft resolution. The template matching code was used to generate two template images for the training set, and the noise variance was increased until there was a noticeable overlap between the distribution of template features on the scatter-plot. This occurred when the noise standard-deviation was about 50. These templates were then used to generate features from the test set, and the same classifier was used to produce a confusion matrix. The method was then repeated with the noise variance set to zero so that the improvement due to the noise robustness could be quantified. Both of these confusion matrices are shown in Figure 6.5. The two templates used to produce these results are shown in Figure 6.6.

The particular subset of INGARA circular-spotlight SAR data considered in this report

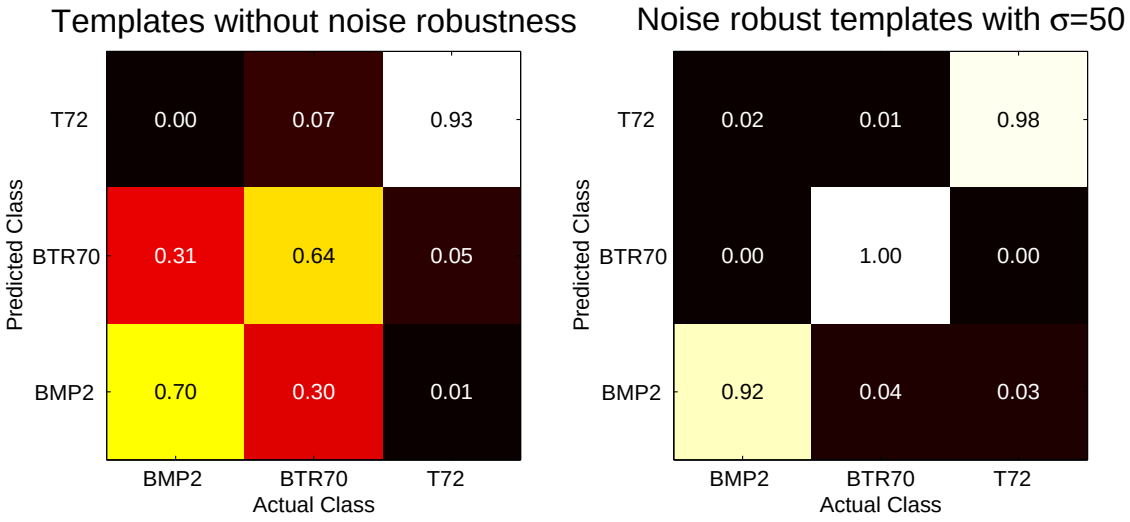


Figure 6.5: *The effect of adding noise robustness to a template matching classifier*

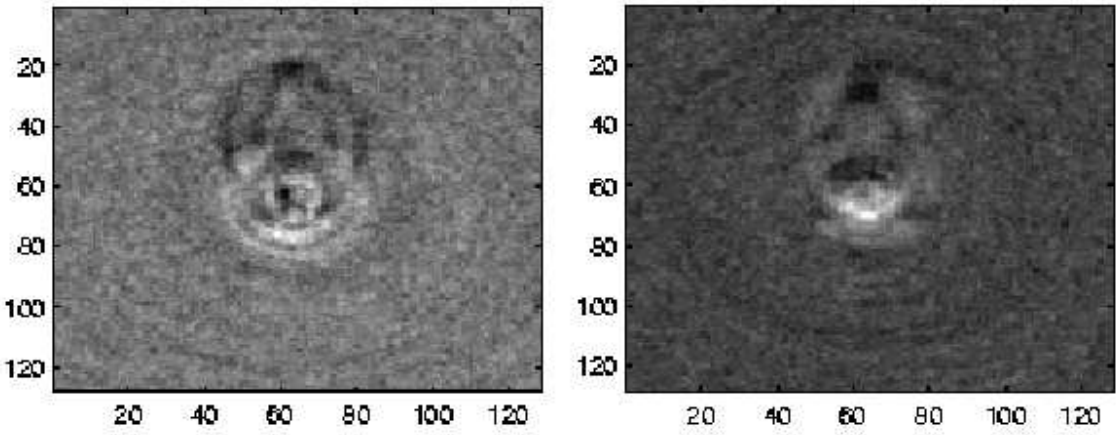


Figure 6.6: *The best templates produced for the MSTAR dataset*

consists of data collected during six different runs, and of four different ships. A total of 22 files were used, with each file containing a spotlight image sequence centered (using the brightest pixel) on a particular target ship. Every sequence was measured from an aircraft which was looping about the target of interest, so different images correspond to different target aspect angles. The 81×81 sized images have been reprocessed however, so that the targets appear to be aligned in the same direction within each image, although due to the strong blurring effect in the azimuth direction, this is difficult to make out. Some examples of the type of imagery being dealt with are shown in Figure 6.7. Training and test image sets were then constructed by splitting the data into two equal sets. Images from the same sequence were assigned to the same data set for classification so that correlations between successive images within a sequence did not bias the classifier.

To test the results of the template matching classifier against those of a standard feature reduction method, Principal Component Analysis (PCA) was applied to the data set to generate 50 eigen-images, the first six of which are shown in Figure 6.8. A simple pairwise linear discriminant was then used to separate the classes in this eigenspace. This led to the confusion matrix of Figure 6.10. In comparison, the template selection scheme from the previous section was also applied to the data, with the unknown noise parameter σ again increased until the training error became non-zero. For this case, the noise was set to $\sigma = 40$. Figure 6.9 shows the templates that were extracted, as well as a scatter plot showing the distribution of each class of the training set within the template image space. The corresponding test set confusion matrix is shown in Figure 6.10, which is a significant improvement over the PCA result. Because of the large amount of azimuth blurring present in the imagery, any given image of a ship can easily contain some of the blurring from near-by ships. Because the ships maintain there relative position in each of the images, the position of this blurring in the image, corresponding to the relative position of neighbouring ships, might be being implicitly used in the template matching. As a result, this classification result may not be representative.

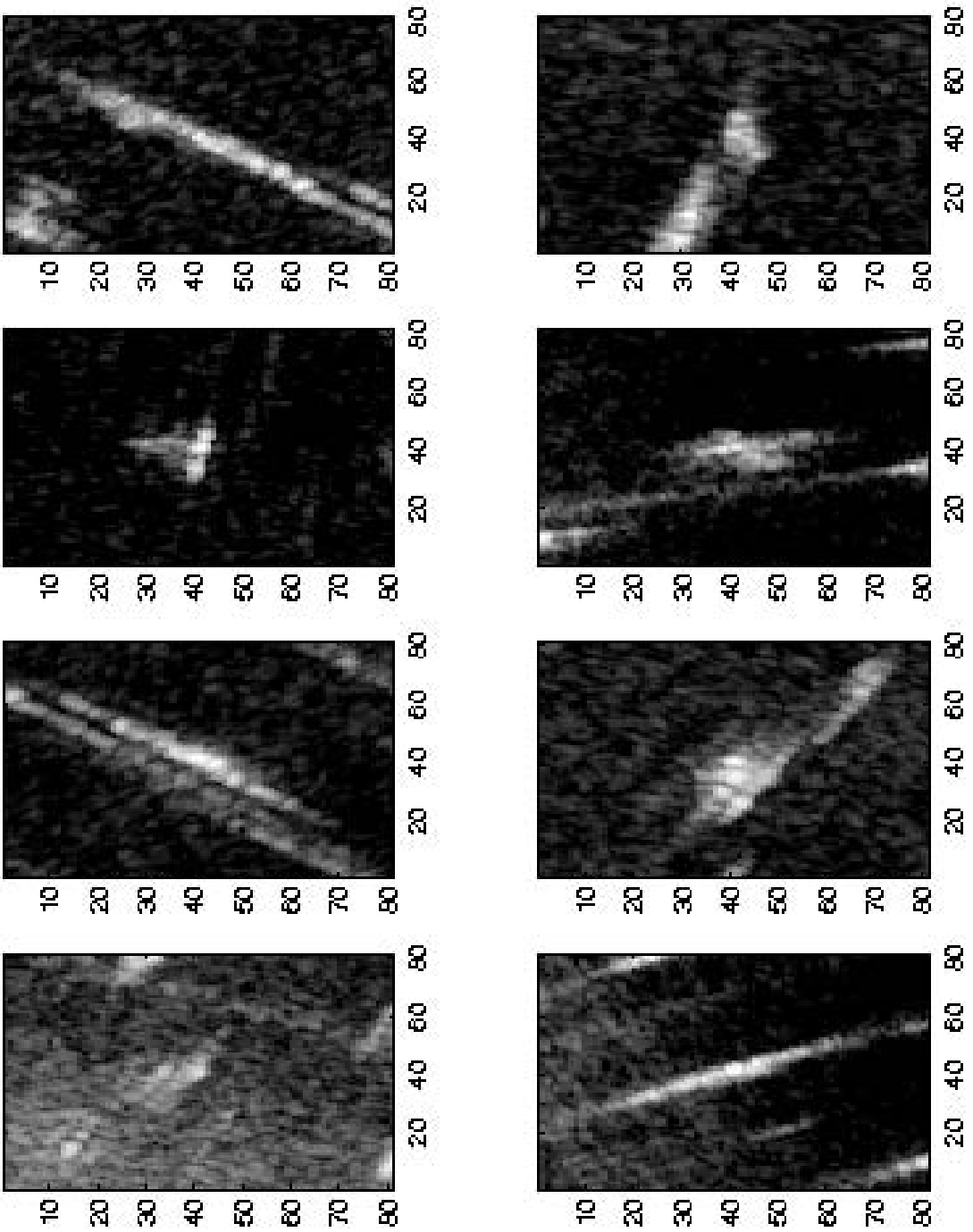


Figure 6.7: *Some examples of images of ships taken using the INGARA radar*

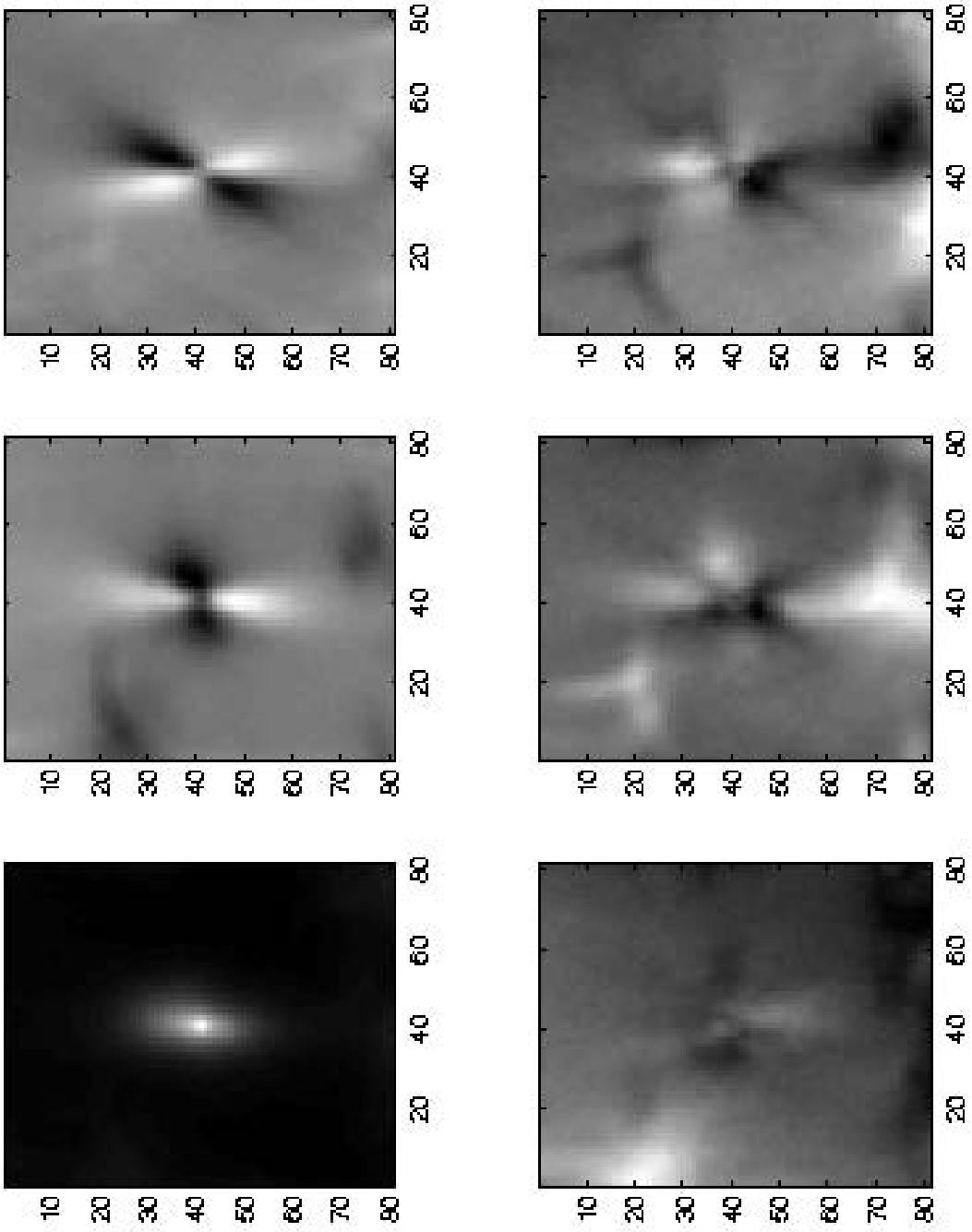


Figure 6.8: *The first six eigen-images for the INGARA data set*

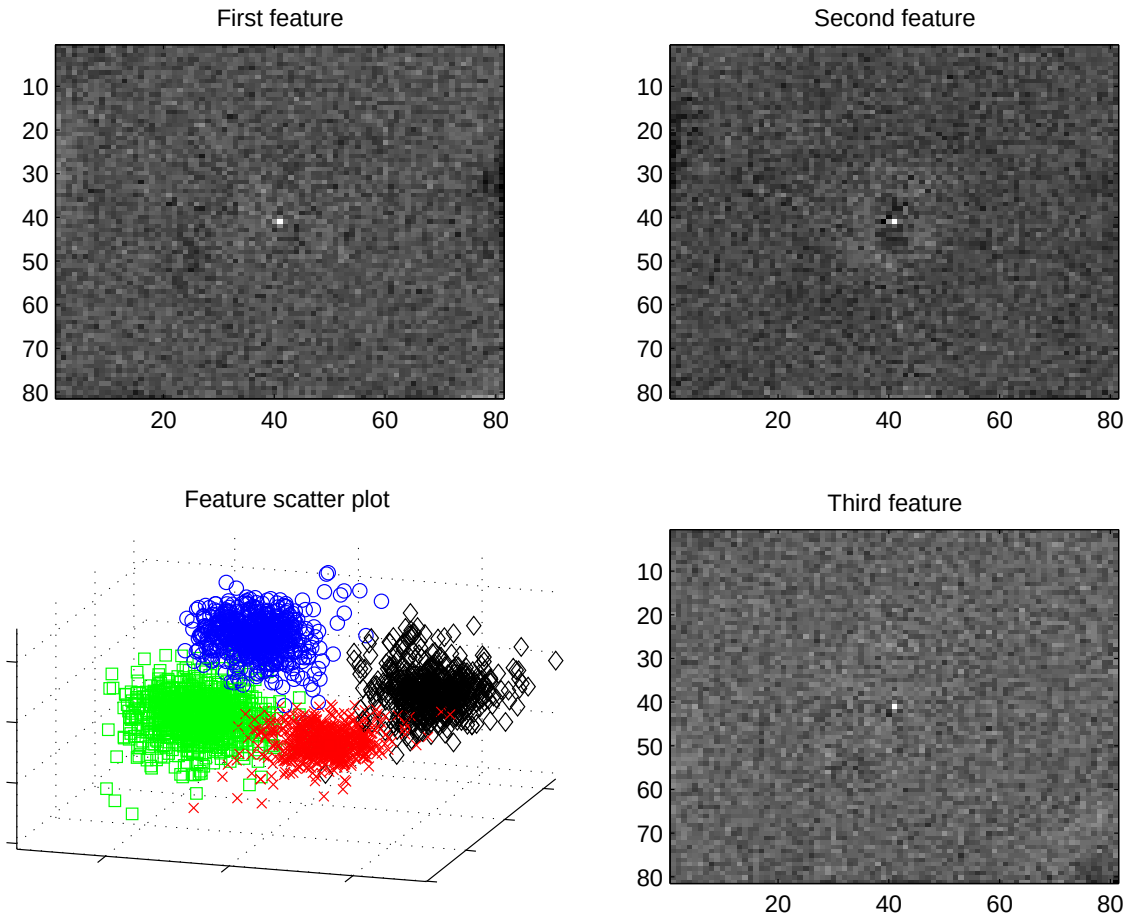


Figure 6.9: *The matched filter templates for INGARA ship data with* $\sigma = 40$

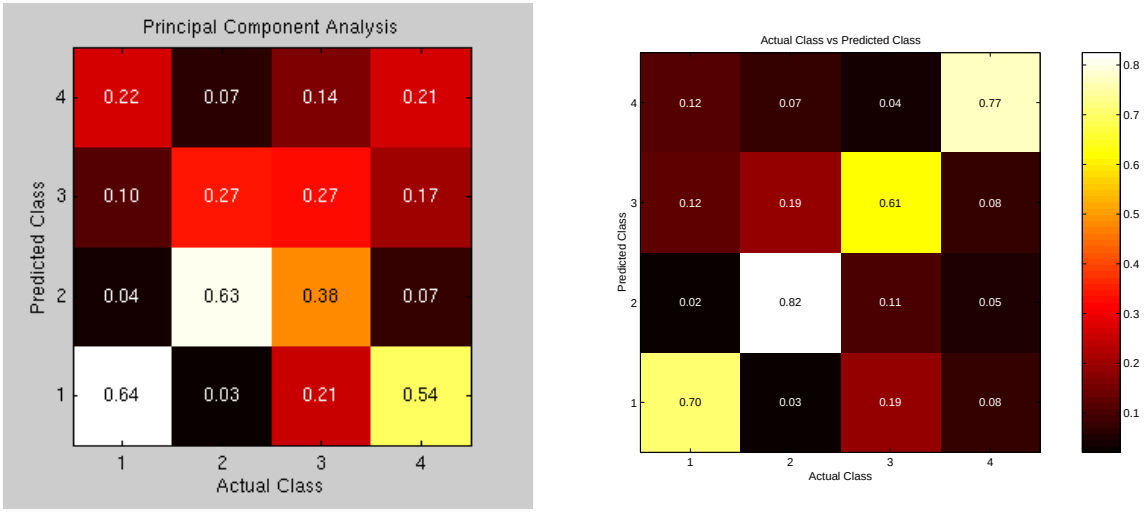


Figure 6.10: *The confusion matrix for the INGARA data with PCA*

6.2.6 Comparison of prescreeners

In this subsection, several of the prescreeners described over the course of this section have been tested on a dataset of real intensity based imagery. For purposes of a comparison with existing results for a large number of prescreeners, the same data set (*i.e.* images from the UK Regular Army Assistance Trial (RAAT) collected by QinetiQ on Salisbury Plains training ground, November 2001) is used here as was used by Blucher et. al. [2]. Although the exact false alarm rates and detection rates are likely to be very different for maritime scenes than for land due to the different characteristic of both background and targets, it is still expected that the relative performances of the different prescreeners will still be similar.

The RAAT data set contains two types of background; open and scrubland. For each of these two backgrounds, targets have been positioned in the scenery, both with and without optical camouflage. For each of these four scenarios, an aircraft was flown in an octagonal trajectory around the target site, and a SAR image was collected over each leg of the journey, leading to a total of 32 images. Since the template matching prescreener requires the use of training data, the data was bisected so that the first four legs of the octagon were used for training, while measurements of all prescreening performance were made using the second four legs from the test set. Since the same numbered leg does not necessarily show the same target at the same orientation, there should not be any sample bias in the training set compared with the test set.

A comparison of a number of these prescreeners over all 16 test images for HH polarisation (results for PWF and full polarisation may be considered in a subsequent report) is shown in Figure 6.11. A more detailed breakdown of the performance of the standard Gaussian prescreener is shown in Figure 6.12. As expected, this shows that it is more difficult to detect targets in more cluttered regions, while optical camouflage has little observable effect against radar.

The Gaussian CFAR detector relies on distributional assumptions about the background pixels. Figures 6.13 and 6.14 were generated for the RAAT radar images by taking

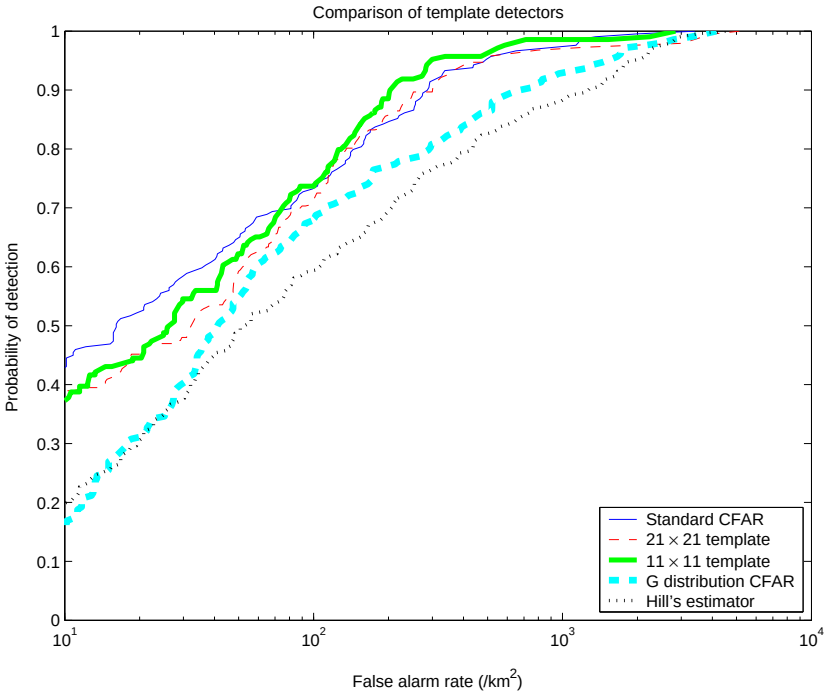


Figure 6.11: *Comparison of the results of various prescreeners*

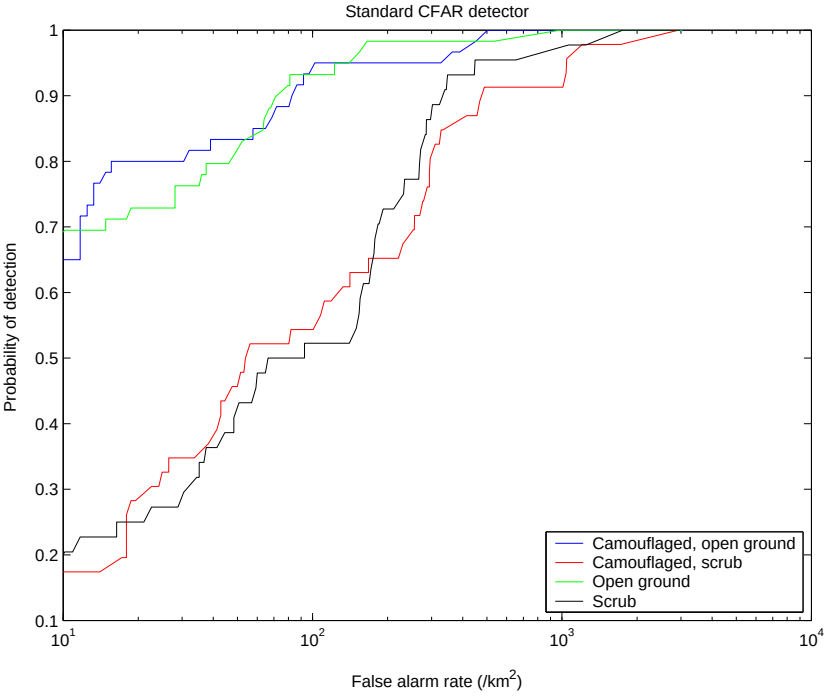


Figure 6.12: *The performance of the standard CFAR for various scenarios*

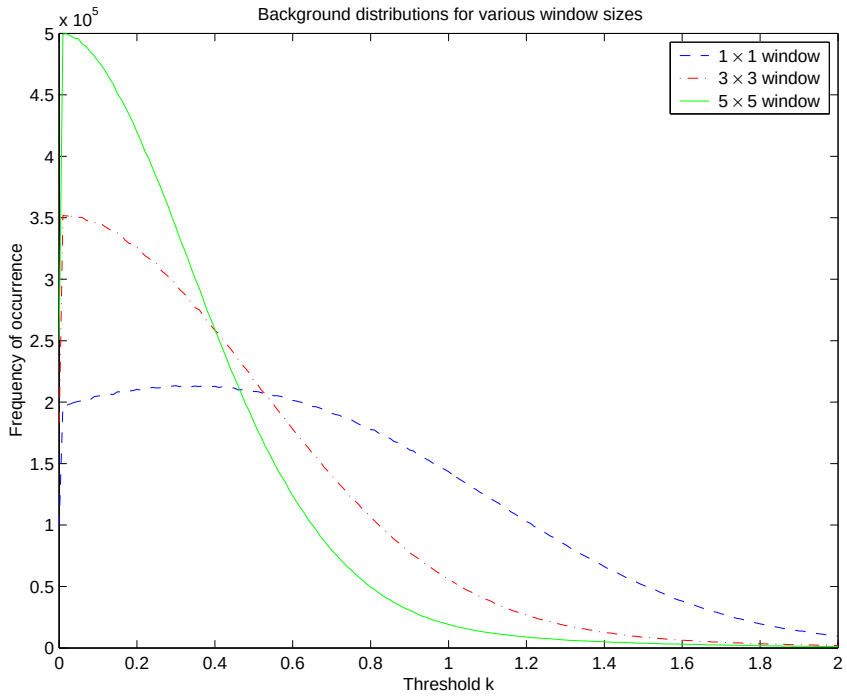


Figure 6.13: *Histograms of the background pixel thresholds*

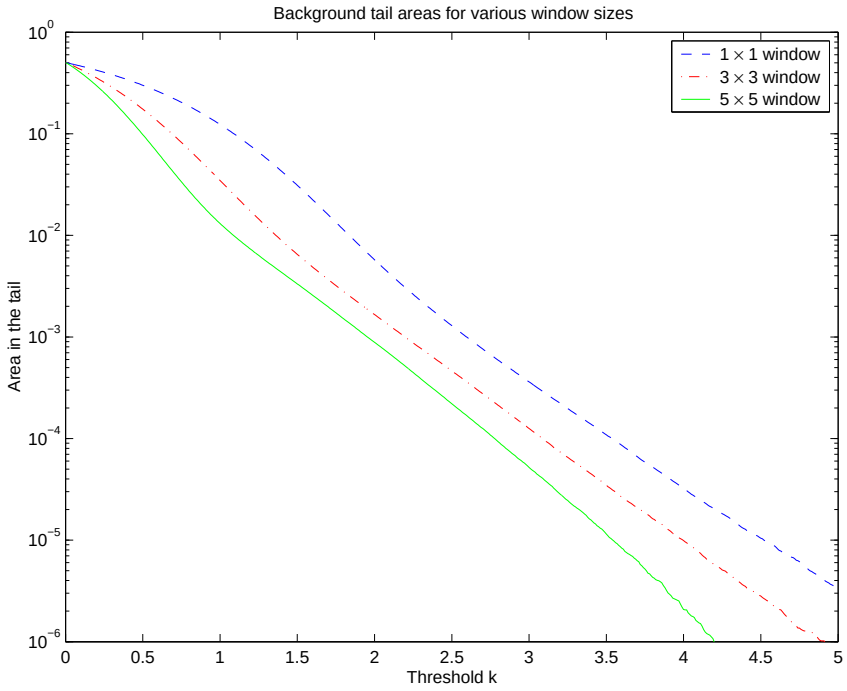


Figure 6.14: *Plot of FAR as a function of threshold for the multiwindow CFAR*

each $n \times n$ pixel signal window, and subtracting its background mean and dividing by its background standard deviation. Figure 6.13 shows the histogram of the mean of the signal window, while Figure 6.14 shows the area in the tail (or the fraction of false alarms) as a function of the threshold k. For fairly large thresholds, the graphs seem very close to linear, which corresponds to an exponential tail rather than a Gaussian. The CFAR detector for an exponential distribution is of the same form as that for a Gaussian distribution, which may account for its improved performance when compared to the K-distribution CFAR detector, as reported by Blucher et. al. [2].

Subsection 6.2.1 described the G distribution, which has been championed by several authors as a good model for the single point statistics of background clutter in SAR images. To test this, the parameters of the G distribution were adaptively estimated using the equations derived in Subsection 6.2.1, and the tail probabilities corresponding to each pixel in the image were then calculated. Figure 6.15 shows a plot of the actual fraction of false alarms against that estimated by assuming G distributed pixels for various values of N, the number of looks. For comparison, these curves have also been plotted for the Gaussian and exponential distributions. For $N = 1$, most of the pixels would not even return a valid estimate for the background parameters, and the fit is very poor. The best fit occurred for $N = 4$, despite the fact that the data was actually single look.

Since $N = 4$ gave the best fit, this is what was used in the CFAR detection results presented as the cyan line in Figure 6.11 and in the more complete breakdown in Figure 6.15. Despite the fact that the statistical model fits better than either the normal distribution or the Gaussian, the standard CFAR detector still provides better results.

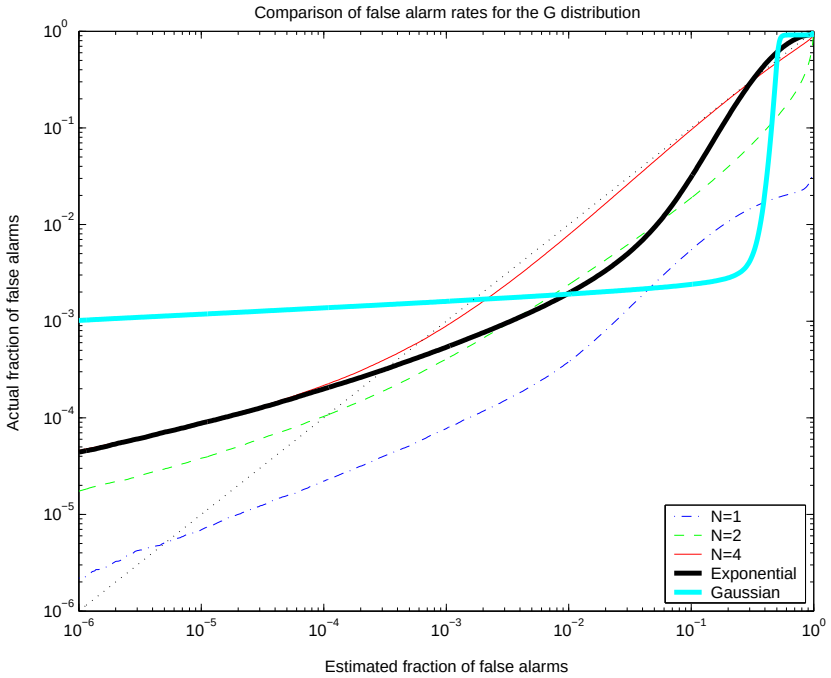


Figure 6.15: *Comparison of actual false alarms against the G distribution estimates*

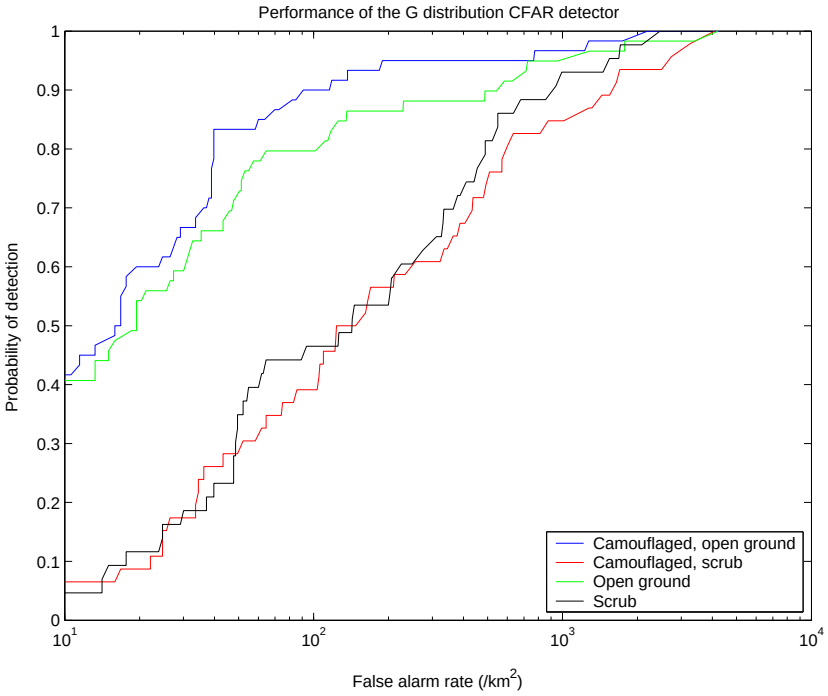


Figure 6.16: *Performance of the G distribution CFAR detector*

The Hill's estimator directly measures the statistics of the tail, unlike the other distributions which will accept errors in fitting the tail distribution in order to better fit the main component of the distribution. To test how accurately the Hill's detector fit the distribution, a histogram of the estimated tail probability for each image pixel was constructed. This histogram was then used to compare the actual tail probabilities with those that were estimated by the Hill's detector. The result of this comparison for the assumption of a 10 percent tail is shown in Figure 6.17. The estimated and actual curves are very close at the 10 percent mark, but this is expected because the estimator is extrapolating based on background measurements about this point. It is also expected that the false alarm rate becomes inaccurate higher than 10 percent, since the form of the distribution assumed is only likely to hold true for the lower false alarm rates. In the tail area however, the estimated false alarm rates have been greatly exaggerated by about two orders of magnitude. The most likely reason for this is that the the ten percent level may still be associated with the main body of the distribution, and that the tail only starts to dominate for larger intensities. This hypothesis was tested by repeating the simulation assuming a two percent tail instead. The results shown in Figure 6.14 show that this gives the most accurate tail statistics of all of the tested methods. Despite this, Figure 6.11 shows that the performance of the Hill's estimator detector, is the worst of the tested prescreeners. A more detailed synopsis of this detector's performance is shown in Figure 6.18. This seems to imply that a more accurate knowledge of the background distribution will not necessarily improve the performance of a prescreener based on single pixel statistics.

The template prescreener from Subsection 6.2.5 was applied to the training data from the first four legs of each octagon in a recursive manner. This was done by starting with

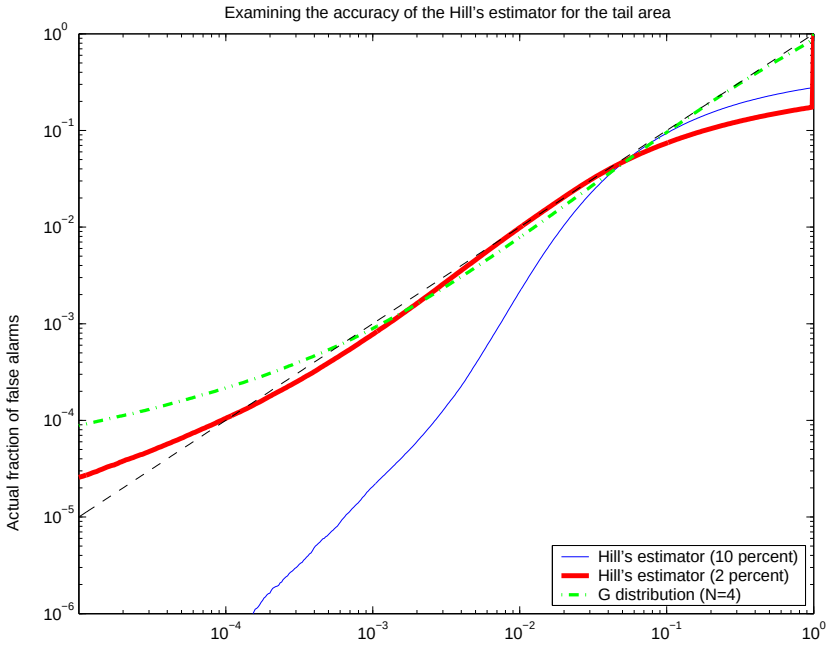


Figure 6.17: *Plot of real versus estimated FAR for the Hill's detector*

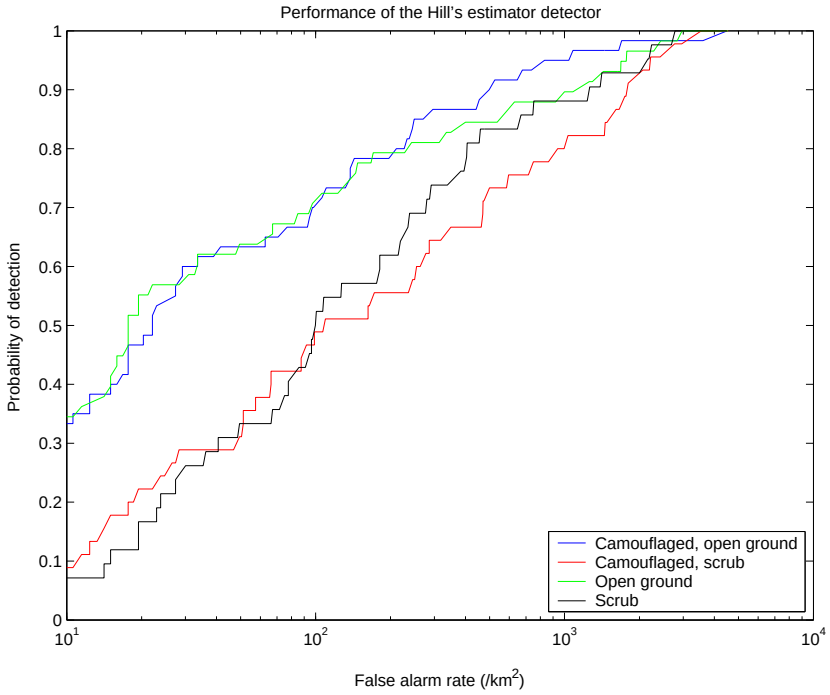


Figure 6.18: *Performance of the Hill's detector for four target scenarios*

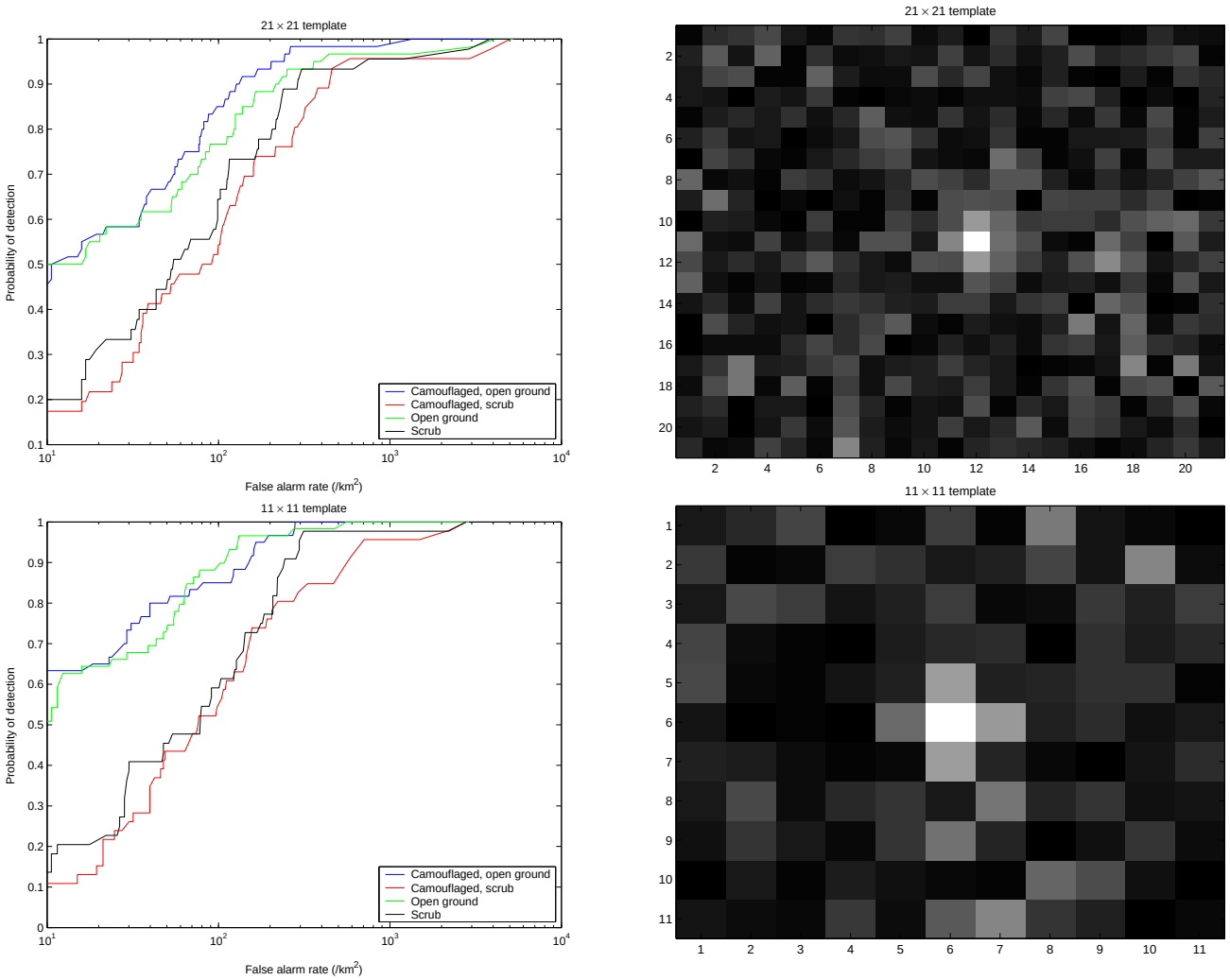


Figure 6.19: *Detailed performance curves for the template prescreener*

a standard CFAR detector template (with a single weight of one in the middle, with the remainder zero), and using this to make detections in the training image so that the probability of detection was about 95 percent. The false alarms were then used along with the strongest template match near the expected target position (which is assumed to correspond to the center of the target) to estimate the correct template weights as described in Subsection 6.2.5. The unknown noise parameter σ was chosen to give the smallest error on a small validation set, which was made independent from the rest of the training data. The procedure was then repeated until it was felt that the template had converged (the point of exact convergence was difficult to determine due to the presence of statistical fluctuations caused by different false alarms being selected for the training step for each iteration). Prescreening performance measurements were made for a number of different window sizes. The results were mostly very similar, so only two are given here in Figure 6.19.

In both of the above cases, 6.11 shows almost equal (or perhaps marginally better) performance to the Gaussian CFAR detector near the operating point for which the template was trained (*i.e.* $90 - 95$ percent PD). This is because the associated template images have the majority of their weight concentrated in the central pixel, just as in the standard CFAR detector. While the standard CFAR detector appears to be somewhat better at for lower PDs, this will not, in general, be of particular interest since the usual operating point for most radar systems is a much higher PD. As was also found by Blucher [2], there does not seem to be a great deal of advantage to using a different prescreener for this imagery. This indicates that there is not any easily discernible structure common to all of the targets in the images. In maritime imagery, the situation may be somewhat different in that the targets will all be moving to some extent, which will result in blurring in the azimuth direction. The fact that there is blurring of targets, but not necessarily of background, may aid in detection, although it does not seem to be possible to use the exact intensity variation along the smear in any form of more complicated object classification (described later in Subsection 6.3.3).

6.3 Low Level Classification

The previous section describes the prescreener, which contains fast algorithms for quickly reducing the input imagery to a relatively small number of possibilities. These detections can be processed with more computationally intensive processes in the low level classifier, which may reduce the overall system false alarm rate still further. The following section describes a few algorithms that have briefly been considered in the course of this contract. A more detailed examination of low level classification algorithms and an analysis of their performance will be examined in the sequel to this report.

6.3.1 Wake detection

One characteristic which can be used to confirm that a detection is actually a target is the presence of a wake. Visual analysis of radar imagery from a number of systems indicates that wakes are more easily discernible in systems with low incidence angles.

This means that wake detection can be helpful in eliminating false alarms from satellite radar images such as ERS. Conversely, for systems such as INGARA which have shallower incidence angles, wake detection proves almost impossible, and is unlikely to improve the overall detection performance.

The following subsubsections describe a number of methods that have been implemented for the measurement of wakes produced by a particular target detection. Each of the methods produce at least one measure of the likelihood or intensity of the wake, and these may be combined with the other features described in this section into a single low level classifier.

6.3.1.1 The Eldhuset ship wake detector

This subsubsection describes the implementation of the Eldhuset wake detector, as outlined in [9]. As with most wake detectors, it is based on finding linear features in the imagery by using the Radon transform. The basic geometry of the detector is shown in Figure 6.20. If a cluster of candidate pixels belongs to a moving ship, then the motion of the ship in range will cause it to appear shifted in the azimuth direction. The vertex of the wake should thus be somewhere along a line in the azimuth direction, but the exact position will depend on the ship velocity, which is not known a-priori. Eldhuset resolves this by testing for the presence of a wake at all possible vertex locations, and then choosing the test position at which the wake seems most intense as the most probable location for the vertex. The actual process of detecting the wake for an individual test point is as follows:

- From the test point, integrate the image pixel intensities along N half-lines directed at angles θ_i $(i \in 1 \ldots N)$ to the azimuth axis. For this report, the integration

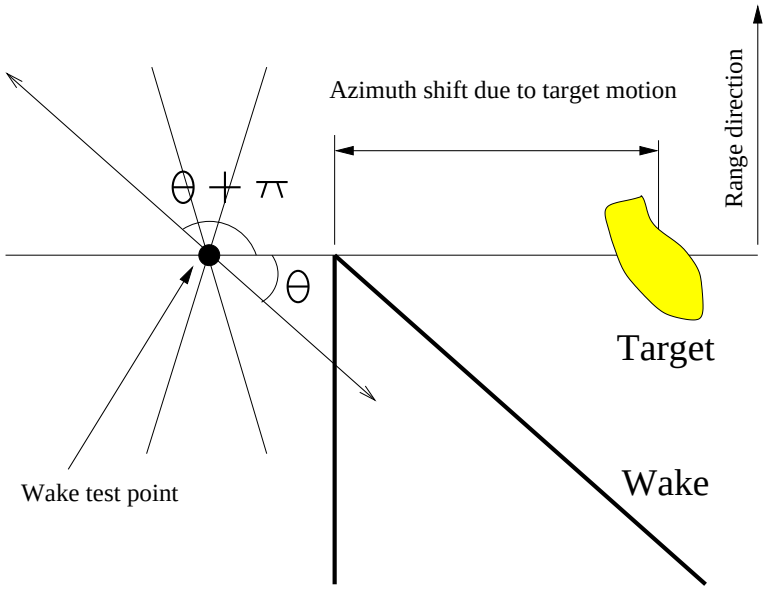


Figure 6.20: *Geometry for the Eldhuset wake detector*

was performed using the standard Radon transform. Since the Radon transform integrates over lines instead of half-lines, the image was first split into two halves (one half with greater range than the target, and the other with less range), and a Radon transform applied to each half individually. From these, the intensity integral $f(\theta_i)$ over the ith half-line can be calculated.

- At the vertex of a wake, there should be at least two half-lines (corresponding to the arms of the wake) at which the intensity should be statistically different from the background. A brighter intensity line might correspond to a Kelvin wake, while a lower intensity line could be a turbulent wake. Eldhuset [9] fits a smooth function $\hat{f}(\theta)$ to the measured $f(\theta_i)$ and assumes that the error is due to statistical fluctuations. By setting a threshold ($k\sigma$ where σ is the standard deviation of the residual, and $k = 3.5$ according to Eldhuset) a wake may be assumed to exist if the residual exceeds this bound. The number of standard deviations of the maximum residual away from the mean will be a measure of the intensity of the wake.

Since wakes are physical phenomenon rather than mathematical constructs, the rules for determining what constitutes a wake are somewhat flexible. The Eldhuset model implicitly lies on two parameters. Firstly, there is the window size over which the Radon transform is calculated. If this is too large, then the linear model for the wake becomes less accurate and detection performance will be decreased. If the size is too small, then random fluctuations in the data will overwhelm the real signal. Secondly, there is the matter of how to accomplish the smoothing. Eldhuset cryptically mentions using a Chebychev approximation to the function $f(\theta)$, but a straight-forward polynomial approximation will not take into account the periodicity of the function, and larger residuals will occur at the end-points of the angle. An alternative might be to write $f(\theta)$ as a sum of even and odd components $f(\theta) = f_e(\theta) + f_o(\theta)$, and then transforming the independent variables so that $g(\cos\theta) = f_e(\theta)$ and $h(\sin\theta) = f_o(\theta)$. Now a polynomial approximation in the transformed domain to the functions g and h will be periodic, so

$$\hat{f}_e = \hat{g}(\cos\theta) = \sum_n A_n T_n(\cos\theta) = \sum_n A_n \cos n\theta$$

and

$$\hat{f}_o = \hat{h}(\sin\theta) = \sum_n B_n U_n(\sin\theta) = \sum_n B_n \sin n\theta$$

where $T_n(x)$ and $U_n(x)$ are Chebychev polynomials of the first and second kind respectively. Therefore the Chebychev approximation alluded to by Eldhuset may be just a simple Fourier approximation. The number of sinusoids required in the approximation is a second parameter that is required to be set. Perhaps the best way to approach this would be to use a training set to determine the best parameters, and to assess the performance of the resulting detector on a separate test set.

6.3.2 Target segmentation

Suppose the prescreener makes a detection, and an image chip centered on this pixel is required to be classified. For this case, the spatial extent and shape of the potential target can be used to determine whether the detection is of interest. This information requires that the potential target information is separated from the background clutter, and this requires segmentation of some kind. This subsection briefly considers a number of different types of segmentation algorithm, while the next subsection describes features based on this segmentation.

6.3.2.1 Region growing

This algorithm assumes that the image to be segmented contains two types of pixels. Those generated by the possible target, which contains the point flagged by the prescreener, and those generated by the background clutter. This assumption may be invalid if there are targets in the image that are close together, since the second target will contaminate the background estimate of the first. It is further assumed that the pixel intensity values for a given class (background or target) are statistically independent. From these assumptions, a region growing segmenter can be implemented using the following steps:

- **Initialise segmentation:** It is known that the centre point belongs to the possible target, and to initialise the algorithm the remaining pixels are all assigned to the background.
- **Determine candidate pixels:** Given the current set of target pixels, work out which other pixels could possibly be from the same target. Usually points from the same target will be fairly close together, so points a certain distance from the current cluster boundary should be considered. These can be obtained fairly simply by performing a dilation of the image using a structural element of the appropriate size.
- **Calculate likelihood ratios:** Suppose that B is the set of background pixels, T is the set of target pixels, and $\{C_i\}$ forms the set of candidate pixels. For any given candidate pixel, there are two hypotheses to consider. In both, $B \setminus C_i$ is produced by the background distribution and T is produced by the target distribution. The null hypothesis is the probability that C_i is generated by the background distribution, which is

  $$P(x = C_i | x \tilde{f_b}(\theta_b))$$

  where f_b is the background distribution with parameters θ_b. Since the parameters are unknown, they must be estimated from all of the sample points (which includes C_i for the null hypothesis).

  The alternative hypothesis is that C_i is generated by the target distribution, which will have likelihood

  $$P(x = C_i | x \tilde{f_t}(\theta_t)).$$

where the target distribution f_t may have a different form from the background distribution, so for instance, a K-distribution may be used to model the background, while the target pixels are modelled by normal distributions. Again, the parameters θ_t must be estimated from the target sample points (which includes C_i for the alternative hypothesis). The likelihood ratio for the ith candidate pixel then becomes

$$\lambda_i = \frac{P(x = C_i | x \tilde{f}_t(\hat{\theta}_t))}{P(x = C_i | x \tilde{f}_b(\hat{\theta}_b))}.$$

- **Boundary constraints:** In general, targets of interest would not tend to have disconnected pixels or long filaments of pixels. To incorporate this prior knowledge into the model, a somewhat arbitrary boundary constraint ratio (similar to that used in traditional segmentation which uses a length term in the Mumford-Shah functional (see for example [13])) has been defined. A compact convex set of points will have a short boundary compared to its area, while a set of disjoint points will have a much larger boundary. This means that in general, a more representative target shape will have a larger ratio 16 $a(T)/l(T)^2$ where a is the area and l is the length. The constant 16 out the front is chosen so that the largest possible value of the ratio (which occurs when T is a square) is one. This ratio can then be multiplied by the likelihood ratio to give

$$t_i = \frac{16a(T \cup C_i)}{l(T \cup C_i)^2} \lambda_i.$$

- **Adding a pixel:** The values t_i indicate how likely a candidate pixel C_i is to be produced by the target region compared with the background region. Suppose $t_j = \overset{max}{i}\ t_i$. Then if t_j is larger than some user defined threshold (about 1 is probably a good choice) then the pixel C_j should be removed from the background B and added to the target set T, and then the process can be repeated. Otherwise the procedure stops.

One down-side to the above algorithm is that long skinny targets (such as ships) will be preferably segmented when they are aligned with the coordinate axis rather than at an angle.

6.3.3 Azimuth smear

Standard SAR processing assumes that the objects being imaged are stationary. Ships, since they are constantly affected by wave action, are constantly changing their velocity. This prevents their being focussed correctly in the azimuth direction, and the target appears smeared in this direction. In many cases, the smear has a much greater extent in the image than the ship itself, so it is of interest to know if any information can be extracted from the smear itself, which can be of use in target classification.

The extent of the smear definitely gives some information about the ship, since it is a measure of the upper and lower velocities of point scatterers belonging to that ship

during the processing interval of the image. If the sea state is known (or can be estimated from the backscatter statistics of the surrounding sea), then an estimate for the height of the ship can be made by assuming that the highest scatterers are the ones responsible for the greatest variation in velocity. Also, the intensity of the ship at that range should be related to the total power contained in the azimuth smear. The question is, can any further information be extracted from the distribution of intensity along the extent of the ship.

Suppose that at each range that a ship contains one scatterer. The only information about this scatterer that can not be determined from the extent or brightness of the azimuth smear is the azimuth position of the scatterer, the motion of the scatterer, or how the scatterer cross-section changes with angle. It's fairly evident that the last will not be possible without also knowing how the reflector changes in angle (*i.e.* its motion), so here it will further be assumed that the amplitude of the scatterer remains constant over the processing interval.

In [7], it was found that motion and position estimates could be extracted from a sequence of ISAR images. In the current problem, a single SAR image is available, which is effectively a coherent sum of a sequence of ISAR images. This summing makes it impossible to determine at what point in time an individual scatterer had any particular Doppler. To highlight this point, a slice with fixed range was taken through one of the blurred INGARA ship images from Subsection 6.2.5.3. The orders in which the measured velocities (as defined by the azimuth smear intensities) occurred were then varied in a number of ways. Figure 6.21 shows some possible motions consistent with an incoherently produced azimuth smear. The plots in the first diagram each have a different frequency while those in the second have an identical frequency, but different phase. Thus there are effectively an infinite number of possible motions for each scatterer. Also, for an individual smear, it is not possible to distinguish between a high scatterer subjected to roll/pitch and a scatterer in the horizontal plane subject to yaw. Hence, if any useful information can be extracted from the smears several independent examples would need to be considered.

When independent azimuth smears are available (which seems exceedingly rare in the INGARA data from subsection 6.2.5.3), they are likely to have slightly different distributions of intensities. In the simple case where it is assumed there is only one scatterer per range bin, this will be because the scatterers have different azimuth positions, so the measured Doppler will contain different components of yaw and roll/pitch velocities. Now three components which contribute to the azimuth smear: the roll/pitch V_R, the yaw V_Y and the translation V_T velocities. Although there is still no way for these to be determined as functions of time, in the special case when all three components are independent and distinctly distributed, there might be enough information to determine them in the spectral domain, and therefore extract positional information about the scatterers in azimuth.

Suppose that the unknown motion components of the target have velocity distribution functions to be determined at M points. Now the intensity within the ith smear will be the distributional addition of each of the rotational components, so

$$A_i(v) = \left(\frac{1}{h_i} V_R\left(\frac{v}{h_i}\right)\right) * \left(\frac{1}{x_i} V_Y\left(\frac{v}{x_i}\right)\right) * T(v)$$

where the $*$s are convolution operators. It is probably easier to solve these equations

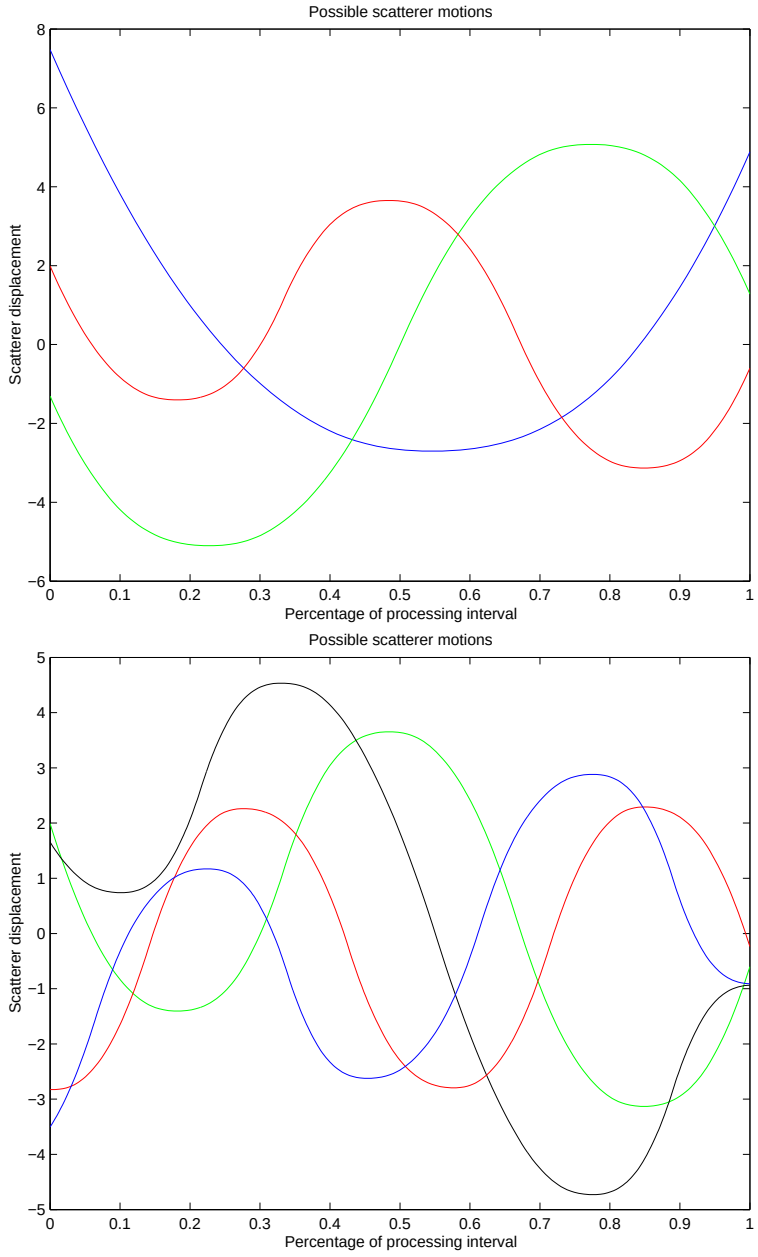


Figure 6.21: *Some plausible motions consistent with a particular azimuth smear*

in the Fourier transform domain, where all of the convolutions become multiplications. Now suppose that N range bins contain azimuth smearing from the same target, and that M' intensity measurements are available within each smear. Then there are NM' known quantities and $3M + 2N$ unknowns, which should, in theory, be solvable for $N >> 3$.

In practice, the rotational velocities should be quite strongly dependent, especially over the comparatively short (in number of wave periods) integration time of most conventional SAR systems. This makes addition of the motion components impossible in any meaningful way. Also, there will be multiple scatterers contributing to the smearing in each range bin. This might still be handled under the above structure if the smearing added coherently, but the addition of a phase difference between the scatterers as a function of time, which would be dependent on the angular orientation of the ship, would make the independence assumptions even less valid. In short, it would be of much more use to reprocess the data as ISAR images, and extract any useful data from that.

References

1. P.Antonik, B.Bowles, G.Capraro, L.Hennington, A.Koscielny, R.Larson, M.Uner, P.Varshney and D.Weiner, “Intelligent use of CFAR algorithms,” Rome Laboratory report, RL-TR-93-75, May 1993.

2. G.Blucher, D.Blacknell, N.Redding and D.Vagg, “ Prescreening algorithm assessment within the Analysts’ Detection Support System ”, ISAR 2003.

3. H.Bottomley, “One tailed version of Chebyshev’s inequality,”

   `http://www.btinternet.com/~se16/hgb/cheb.htm`

4. T.Cooke and M.Peake, “The optimal classification using a linear discriminant for two point classes having known mean and covariance,” Journal of Multivariate Analysis, Vol.82, No.2, pp 379–394, August 2002.

5. T.Cooke, “First report on features for target/background classification,” CSSIP-CR-9/99, April 1999.

6. T.Cooke, “Third report on features for target/background classification,” CSSIP-CR-5/00, May 2000.

7. T.Cooke and D.Gibbins, ”ISAR 3D shape estimation,” CSSIP CR-15/02, June 2002.

8. T.Cooke, N.Redding, J.Zhang and J.Schroeder, “Target detection survey,” CSSIP-CR-11/99.

9. K.Eldhuset, “An automatic ship and ship wake detection system for spaceborne SAR images in coastal regions,” IEEE Transactions on Geoscience and Remote Sensing, Vol.34, No.4, July 1996.

10. B.M.Hill, “A simple general approach to inference about the tail of a distribution,” Annals of Statistics, Vol.3, pp 1163–1174, 1975.

11. L.M.Novak, S.D.Halversen, G.J.Owirka and M.Hiett, "Effects of polarization and resolution on SAR ATR," IEEE Transactions of Aerospace and Electronic Systems, Vol.33, No.1, January 1997.

12. N.J.Redding, D.I.Kettler, G.Blucher and P.G.Perry, "The Analysts' Detection Support System: Architecture Design and Algorithms," DSTO-TR-1259, July 2002.

13. N.J.Redding, D.J.Crisp, D.Tang and G.Newsam, "A comparison of existing techniques for segmentation of SAR imagery and a new efficient algorithm," Proceedings of DICTA'99, pp 35–41.

14. N.J.Redding, "Estimating the parameters of the K-distribution in the intensity domain," DSTO-TR-0839.

15. J.S.Salazar, "Detection schemes for synthetic aperture radar imagery based on a *beta prime* statistical model," Proceedings of conference on Information Systems, Analysis and Synthesis, 2001.

16. A.Sato, "Discriminative dimensionality reduction based on generalised LVQ," ICANN 2001, pp.65-72, 2001.

17. C.C.Wackerman, K.S.Friedman, W.G.Pichel, P.Clemente-Colón and X.Li, "Automatic detection of ships in RADARSAT-1 SAR imagery," Canadian Journal of Remote Sensing, Vol.27, No.5, October 2001.

Chapter 7

Low Level Classification

7.1 Introduction

Maritime detection, like many other detection tasks, consists of three main components. The first step is prescreening, which consists of a simple and fast algorithm for extracting likely detections from an image. This stage usually has a high detection probability, and a similarly high false alarm rate. The advantage, however, is that prior to prescreening the targets of interest could appear anywhere in the image, whereas afterwards the focus of more computationally intensive classifiers is only on a relatively small number of points. Prescreening was the focus of a previous report [5].

The next component of detection is the low level classifier. This uses more computationally intensive techniques for processing local image information from each of the prescreened points, to remove more false alarms. The final stage, which is frequently absent, is high level classification, which uses global structures within the image (positions of coast, shoals, islands, weather conditions, etc.) to determine how likely each of the remaining detections are to be real targets.

The subject of the current report is the low level classifier. Due to the similarity of techniques for low level classification in different applications, and the small amount of useful maritime data available, the emphasis of the report is on general methods. These methods fall into two categories. The first is feature extraction, which is the subject of Section 7.2. This is where an image containing a detection from the prescreener, described by a large number of pixel intensities, is reduced to a lower dimensional feature space. The ideal feature space would throw away data that is not useful in separating targets from background clutter (such as pose, or translation errors), so much of this section describes rotation and translation invariant features. The next category is classification, where rules for determining which images are of interest are determined based on the features space. Similar methods have been reported on in previous reports [9], and so Section 7.3 of the current report has outlined some of the newer work on ensemble classifiers (such as boosting and bagging) which had not yet been covered in depth. Finally, some conclusions are concerning the use of these general features and classifiers in maritime surveillance problems have been expounded in Section 7.4.

7.2 Feature Extraction

Classification of images is often a very difficult problem, since most techniques suffer from the "curse of dimensionality". This is where the addition of dimensions containing no useful information allows the training data to be separated better, but reduces the ability of the classifier to generalise to examples not in the training set. Images are usually very high dimensional, with the number of dimensions corresponding to the number of pixels. Therefore in order to improve classification, often a small number of features are extracted, and the classifier is applied to the features. This section describes a number of experiments relating to feature extraction.

The layout of this section is as follows. In Subsection 7.2.1 the RAAT data set, used in the later experiments, is described. Subsection 7.2.2 then outlines a simple method for linearly combining polarisations into a single feature. Then in Subsection 7.2.3, an implementation of the Radon transform using a non-equidistant Fourier transform is discussed, and several modifications have been proposed to make it translation and rotation invariant. The performance of linear templates based on these transforms have also been evaluated. Some other invariant features have also been discussed.

7.2.1 Data sets

Due to the absence of suitable maritime imagery, the performance of the various low level classification algorithms have been measured using the RAAT data set. This is a set of 32 spotlight SAR images of a land area, each containing ten to twenty vehicles, whose position within the images are known. Each pixel within the images corresponded to 0.3×0.3m on the ground.

The 32 RAAT images form four groups of eight images. Each of these groups corresponds to the same target positions imaged from a different direction, since the aircraft flew along eight sides of an octagon, with each image corresponding to a different side. The four groups can be categorised according to the difficulty of detecting the target, and consist of either camouflaged or uncamouflaged vehicles in either open ground or in scrub. The scrub data set contains large numbers of occluded targets, which are unlikely in maritime imagery. They have been included, however, to allow comparison with earlier prescreener results on the same data set [5, 2]. In hind-sight however, these should probably not have been used, as will be discussed later.

The experiments in this section are based on a set of training and test images, which are centered on detections produced by ADSS using the ATA prescreener with a threshold of 6.5, followed by a clustering stage with cluster distance of 20 pixels. The training set was extracted using targets and ATA detections from the first four octagon edges for each image group. In some of the experiments, a cross-validation set, consisting of a third of these images, is randomly chosen from this set. The test set contains the targets and false alarms from the remaining four images in each group.

7.2.2 Polarimetric templates

One of the standard methods for processing polarimetric SAR imagery is the PWF (polarimetric whitening filter) [22], which coherently combines the polarimetric channels so that the intensity of man-made objects (having a particular polarimetric model) appear significantly enhanced. The alternative proposed here is a linear template in both space and channel (*i.e.* the equivalent of three linear filters; one for each channel). This has been implemented using the regularised discriminant from the previous report [5] to generate the template for a detection probability of 90 percent. The regularisation made the discriminant more robust to the small sample size by artificially increasing the size of the training set using existing images corrupted by white noise. In the previous report, linear templates were calculated for a set of targets and background in some single polarisation HH data, and it was shown to give slightly better discrimination than the standard ATA algorithm. This procedure has now been repeated for multi-polarisation imagery.

The template was trained on chips from ATA processed images of each of the polarisations. Each chip was either centered on a target, or a false alarms detected using ATA using the HH image only. The template was then applied to the test image set to combine the ATA processed polarimetric channels into a single spatial image. The results of this experiment are shown in Figure 7.1 and show an approximate reduction in the false alarm rate by 25 percent at the required 90 percent reduction rate. This is not an enormous reduction, but seems to be of a similar magnitude to the reduction achieved by Blucher et. al. [2] on the same data set, with the application of the polarimetric whitening filter.

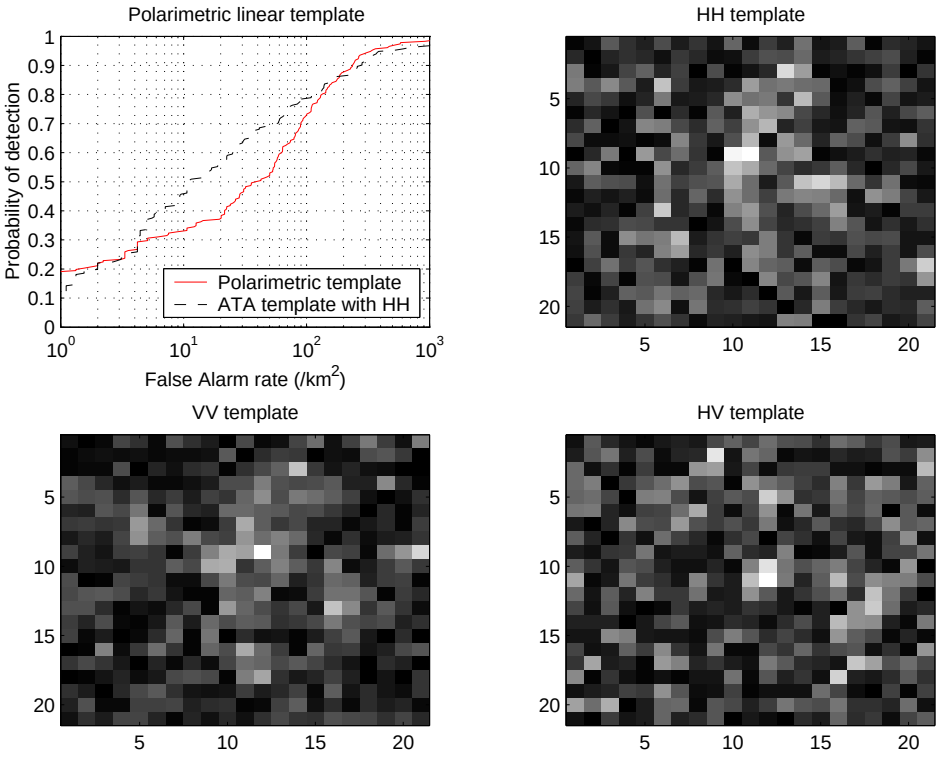


Figure 7.1: *Templates and ROC curve for the linear polarimetric template*

7.2.3 Invariant features

Matched templates, such as those discussed for polarimetric imagery in the previous subsection, can give very good performance when each target of the same class produces a similar image. In general however, targets will be oriented differently, or may not be correctly centered in the image window. To reduce the effect of translation, rotation and scale, it is considered useful to classify based on features which are invariant to these parameters. These also have one of two possible side-effects. In the first case, the invariant features will discard important information (such as with most invariant transforms which usually discard important Fourier phase information), which makes them much less sensitive to differences in image shape. In the second case, the invariant features are so sensitive to image shape that minor variations within the same class of image (not related to rotation, translation or scale) result in large changes to the feature, so the within class variance is not really reduced. This subsection describes a number of different invariant features. First, two types of invariant related to the Radon transform are described in 7.2.3.1 and 7.2.3.2, and their use in classification investigated. Then finally, some of the work on invariant features in optical imagery and other invariant features are summarised in 7.2.3.3.

7.2.3.1 Rotation invariant transform

The projection slice theorem states that the Radon transform of a two dimensional image can be calculated by taking the 2D Fourier transform, resampling onto a polar grid, and then taking the 1D Fourier transform in the radial direction. This can be implemented without the need for an intermediate polar resampling step by the use of a non-equidistant Radon transform, as in [17]. Now if instead, the final 1D Fourier transform is taken in the angular direction, and the absolute value taken, the resulting transform will be invariant to rotation. Mathematically, the transform can be written as

$$\begin{aligned}
T(\rho,\omega) &= \frac{1}{2\pi}\int_{\theta=-\pi}^{\pi} F(\rho,\theta)\exp(-j\omega\theta)d\theta \\
&= \frac{1}{(2\pi)^3}\int_{\theta=-\pi}^{\pi}\left\{\int_x\int_y f(x,y)\exp(-j(\rho\cos(\theta)x+\rho\sin(\theta)y))dydx\right\} \\
&\qquad\qquad \exp(-j\omega\theta)d\theta \\
&= \frac{1}{(2\pi)^3}\int_x\int_y f(x,y)\int_{\theta=-\pi}^{\pi}\exp(-j(\rho\cos\theta x+\rho\sin\theta y+\omega\theta))d\theta dydx \\
&= \frac{1}{(2\pi)^3}\int_x\int_y f(x,y)K(x,y,\rho,\omega)dydx
\end{aligned}$$

where the 2D kernel function $K(x,y,\rho,\omega)$ is given by

$$K(x,y,\rho,\omega)=\int_{\theta=-\pi}^{\pi}\exp\left(j\rho\sqrt{x^2+y^2}\cos(\theta+\phi)\right)\exp(j\omega\theta)d\theta.$$

where the phase shift is $\phi=\pi+\arctan(y/x)$. Substituting $\psi=\theta+\phi$ gives

$$K(x,y,\rho,\omega) = \exp(-j\omega\phi)\int_{\psi=-\pi+\phi}^{\pi+\phi} \exp\left(j\rho\sqrt{x^2+y^2}\cos\psi\right)(\cos(\omega\psi)+j\sin(\omega\psi))d\psi.$$

Now since the original angular function was periodic, the Fourier transform will be discrete, and non-zero only when ω takes on integer values. Therefore, the entire integrand will also be periodic, and the integral can be written over the interval $[-\pi,\pi]$ instead of $[-\pi+\phi,\pi+\phi]$. Also, the component containing $\sin(\omega\phi)$ will be antisymmetric, which means that its interval will vanish. This leaves the symmetric component which is

$$\begin{aligned}
K(x,y,\rho,\omega) &= 2\exp(-j\omega\phi)\int_{\phi=0}^{\pi} \exp\left(j\rho\sqrt{x^2+y^2}\cos\psi\right)\cos(\omega\psi)d\psi \\
&= 2\pi j\omega\exp(-j\omega\phi)J_\omega(\sqrt{x^2+y^2}\rho) \\
&= 2\pi\exp(-j\omega(\arctan(y/x)+\pi/2))J_\omega(\sqrt{x^2+y^2}\rho) \\
&= 2\pi(-j\exp(j\arctan(y/x)))^\omega J_\omega(\sqrt{x^2+y^2}\rho) \\
&= 2\pi\left(-j\cos(\arctan(x/y))+\sin(\arctan(-x/y))\right)^\omega J_\omega(\sqrt{x^2+y^2}\rho) \\
&= 2\pi\left(\frac{y-jx}{\sqrt{x^2+y^2}}\right)^\omega J_\omega\left(\sqrt{x^2+y^2}\rho\right)
\end{aligned}$$

where the second line uses Equation 9.1.21 from Abramowitz and Stegun [1] and J_ω is the Bessel function of the first kind, with order ω. Substituting this expression back into the transform, and converting to polar coordinates gives

$$\begin{aligned}
T(\rho,\omega) &= \frac{1}{(2\pi)^2}\int_{x=-\infty}^{\infty}\int_{y=-\infty}^{\infty} f(x,y)j\exp(j\varphi)J_\omega(R\rho)dydx \\
&= \frac{1}{2\pi}\int_{R=0}^{\infty}\left(\frac{1}{2\pi}\int_{\varphi=-\pi}^{} \pi f(R,\varphi)e^{-j\omega\varphi}d\varphi\right)RJ_\omega(R\rho)dR \\
&= \mathbf{H}_{R,\omega}\mathbf{F}_\varphi f(R,\varphi)
\end{aligned}$$

which means that the new transform is equivalent to a Fourier transform in the angular direction (providing the transform rotation invariance) followed by a Hankel transform of order ω in the radial direction. The Hankel transform term does not offer any obvious advantages to the classifier, except that the use of the non-equidistant Fourier transform might allow the radial to polar resampling to be performed more quickly and with less error. Figure 7.2 shows some examples of the transform applied to three different types of rectangles. Rotating each of these rectangles about the centre of the image had no discernible effect on the resulting transform. The last two rectangles differ only by a translation, although the resulting transforms are still quite different.

As an objective measure of the usefulness of the rotational invariant transform in classification, a similar experiment to that described for the polarimetric template was conducted in the transform space for HH imagery instead of the image space for all three

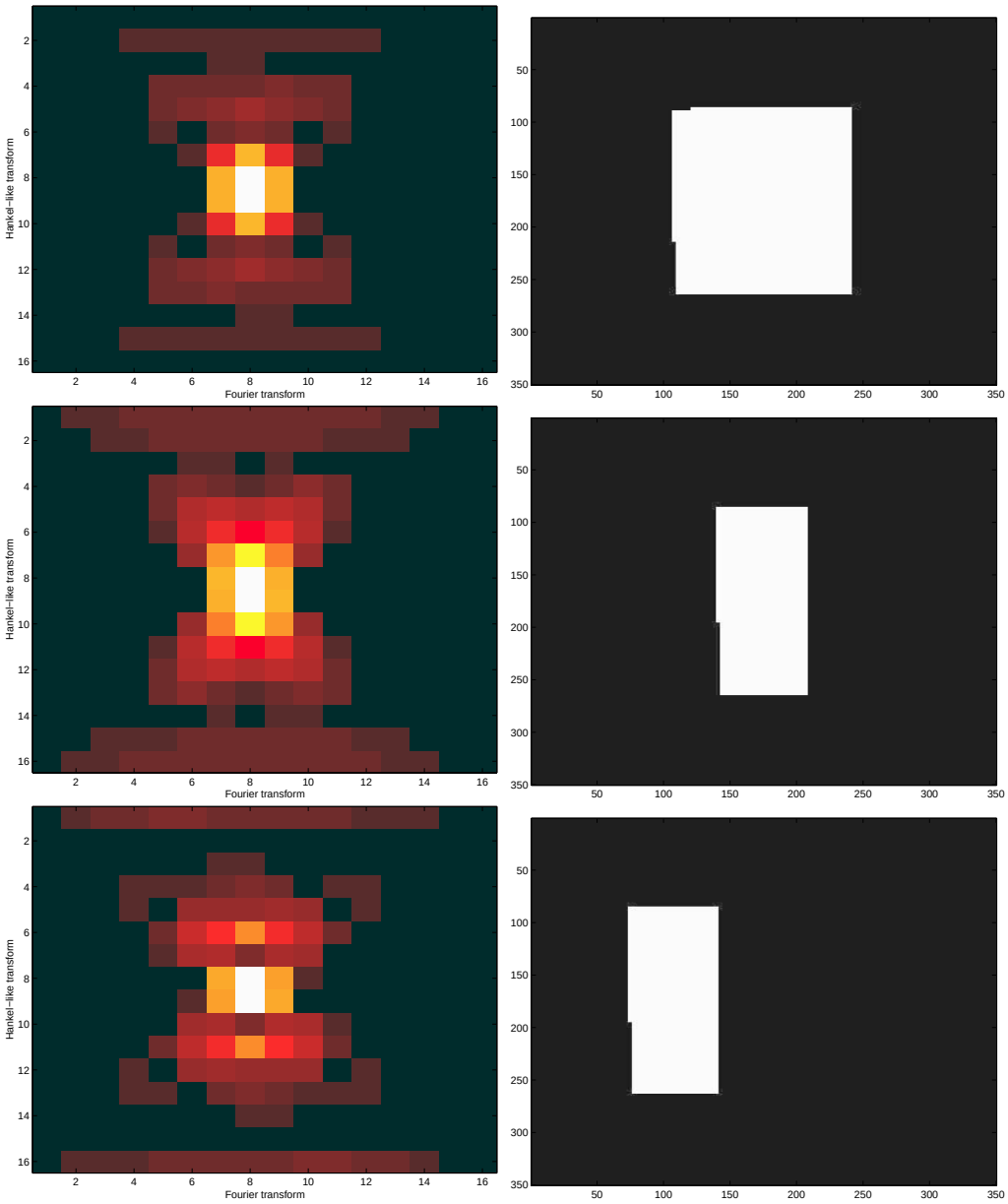


Figure 7.2: *Rotation invariant transform for three types of rectangles*

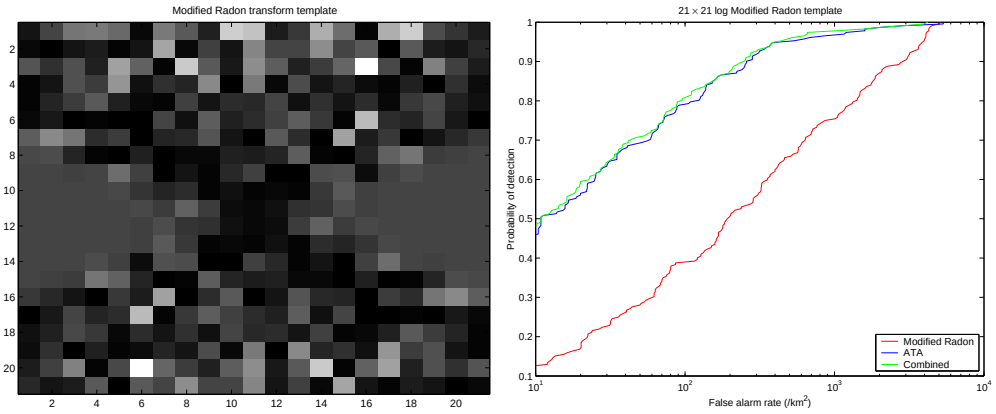


Figure 7.3: *The template and ROC curve for the modified Radon transform*

polarisations. Figure 7.3 shows the resulting transform template and the classification performance based on that template. By itself, the template shows fairly poor discrimination when compared with the prescreener. Even when the template values are combined, the resulting ROC curve is not significantly improved over using ATA alone.

7.2.3.2 Rotation and translation invariant transform

The previously described transform produced rotational invariance by throwing away angular phase information. Radial phase information was retained, but as a result the transform was not translation invariant (as can be seen from the last two examples from Figure 7.2). Translation invariance can be achieved in addition to rotation invariance by taking the absolute value of the 2D Fourier transform prior to the 1D angular Fourier transform. The resulting invariant feature can then be used to construct a feature template for discriminating targets from background, as discussed previously. Figure 7.5 shows the

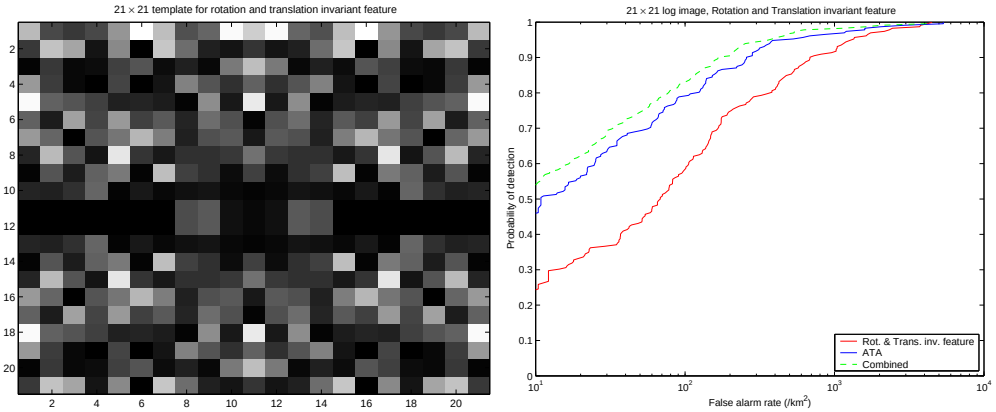


Figure 7.4: *The template and ROC curve for the rotation and translation invariant feature*

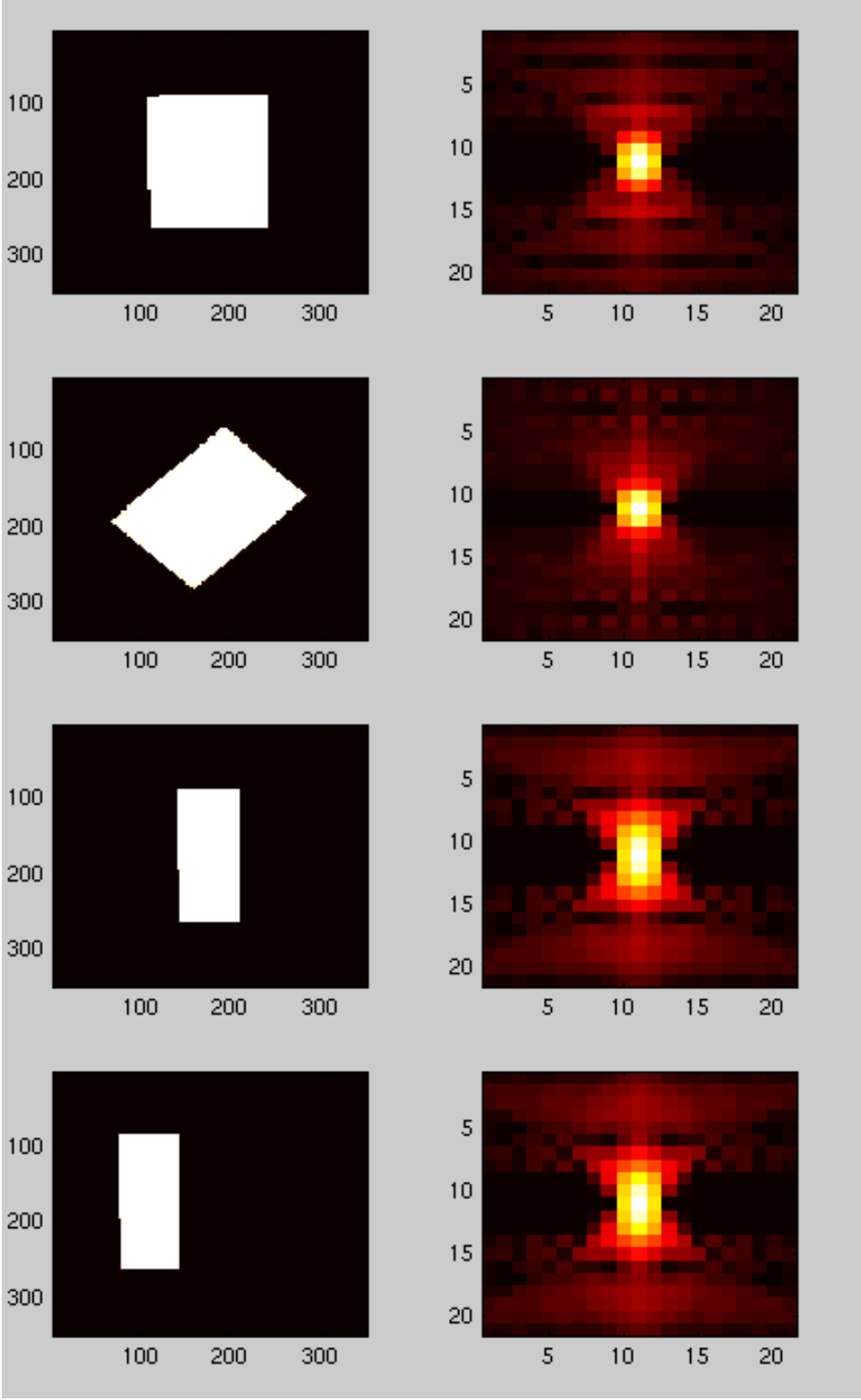


Figure 7.5: *Rotation and translation invariant transform for some rectangles*

output of this transform for four examples with two different types of rectangle. The first two examples show the same rectangle, where one is rotated. The output invariants show some noticeable, although not large, differences. This is due to pixelation error between the two input images. The last two examples are exact translations of each other, and produce identical invariant transforms.

The ROC curves in Figure 7.4 show that the template for this invariant transform reduces the FAR by about one third while retaining a PD of 90 percent. This indicates that this template feature might be useful in classification, although to test this thoroughly the template feature would need to be combined with other features, as described in the next subsection.

7.2.3.3 Other invariants

There are other transformations to which the integral transforms, discussed previously, might be made invariant. For instance, in ISAR processing the same ship undergoing motions related by a scalar multiple would appear to be scaled differently in Doppler. Therefore, the Mellin transform (see for example [15]) has been adopted to produce scale invariant quantities for use in ISAR classification. Scale invariance could also be incorporated into the above integral transforms for SAR by applying a 1D Mellin transform in the radius direction. In SAR imagery however, the pixels correspond to fixed areas on the ground, and since the targets do not vary greatly in size, the scale of the targets will be similar. This means there will not be a large reduction in the target class variance due to the invariance property, but the added step would further increase the sensitivity to minor variations in shape, and the overall class variance would increase. Ideally, a transform would be invariant to target pose and obscuration, but such a transform is not easy to find.

An alternative to integral transforms for invariance has been discussed intensively in the optical imagery literature. The following few paragraphs briefly discuss a selection of the proposed optical invariants. Many of these can also be developed for the SAR image domain, but because they frequently rely on reference points on the target, or the target's boundary to be measured with a relatively high accuracy, they are generally unsuitable for SAR target detection. This is especially true in the maritime domain, where the movement of the target on the waves will often produce a pronounced blurring of the image.

Amongst the papers dealing with the target boundary are Persoon and Fu [23], who introduced Fourier descriptors. This is where the x and y locations of a point on the boundary are plotted as a function of arc-length, and expressed as a Fourier series. The resulting representation is translation and rotation invariant. A similar representation is where the curvature is plotted versus arc-length, such as in Mokhtarian and Mackworth [20] and Lei et. al. [18]. The first paper smoothes the curve using a Gaussian of various widths to obtain a multi-scale representation, while the second fits the curve with a polynomial spline, and compares portions of the boundary using moment invariants. For SAR data however, this representation is worse than Fourier descriptors due to the difficulty in obtaining an accurate measure of the curvature.

Another commonly used invariant for optical imagery is the polynomial moment invariant. Several of these (such as the Hu moments [16]) have been derived, and can be

used to provide pose invariant representations based on a consistently chosen set of points. Again, with SAR imagery, it is difficult to consistently choose points belonging to a target. In fact in one study (reference lost), it was found that even when comparing large and relatively well defined areas within a SAR image the pose invariant moment chosen was so sensitive to shape that the classification was only very modest. For smaller SAR targets, this problem would be expected to get much worse.

In the SAR domain, there has also been a lot of literature related to finding features for use in automatic target recognition. Although invariance was usually not a primary goal, most of them had this property. Some previous work on INGARA imagery (see [6, 7, 8]) describes a large number of features. These features can be broadly categorised as based on histograms, rank statistics, fractal dimension, texture, and matrices (SVDs and similar). Of these categories, only the matrix based features are not necessarily invariant to any obvious transformations and of the remaining features, only the texture based ones are not fully rotation invariant, since they are based on co-occurrence matrices extracted only in the two principal directions. Although the INGARA imagery was at best 1m resolution, the classifier obtained by combining all of these features gave a false alarm rate of about one per square kilometer, with an improvement of about two orders of magnitude over the standard Gaussian prescreener.

In order to compare the performance on the current data set with that obtained previously, all of the features were recalculated for the new data set. The first step in this experiment was to generate feature vectors for each of the individual images in the training set. Due to the difference in image resolution, the features which were computed over 7×7 image chips were modified to use 21×21 sized blocks. Some of the INGARA features (such as the fractal dimension) could not be easily adapted for a 21×21 block from the existing implementation. Also, some features (such as all of the elements of the ranked intensity) became very large, and so were subsampled prior to inclusion in the feature vector. Also, many of the features contained fiddle factors, or were in some way sensitive to the intensity of the image. This was mitigated slightly by calculating the mean and intensity of the local background of each chip, and scaling the central image prior to calculating the features, but no great effort was spent in tuning the features. The result of this was that for each 21×21 image in the training set, a feature vector of dimension 153 was computed. A random one third of the training set was then recruited for use as a cross-validation set.

After the features were calculated, a simple forwards selection scheme was used to determine a subset to be used for classification. Initially $F = \{\}$ was the set of features, and in each iteration, all of the features were tested to see which gave the best classification (using a recursive Fisher discriminant) in combination with the existing set. That best feature was then added to the selected set F, and the next feature selected until the classification error on the cross-validation set started to increase. The exact set of features chosen is highly sensitive to a number of factors, such as the chosen point on the classifier's operating curve. Figure 7.6 shows a number of different ROC curves obtained when the specified probability of detection is changed. As can be seen from this figure, the number of features selected varies from 4 to 22.

For the classifier trained to 90 percent probability of detection, the test error ROC curve was calculated, and this is shown in Figure 7.7. It is apparent that the feature based curve is nowhere near as good as a simple ATA prescreener, which is a quite marked

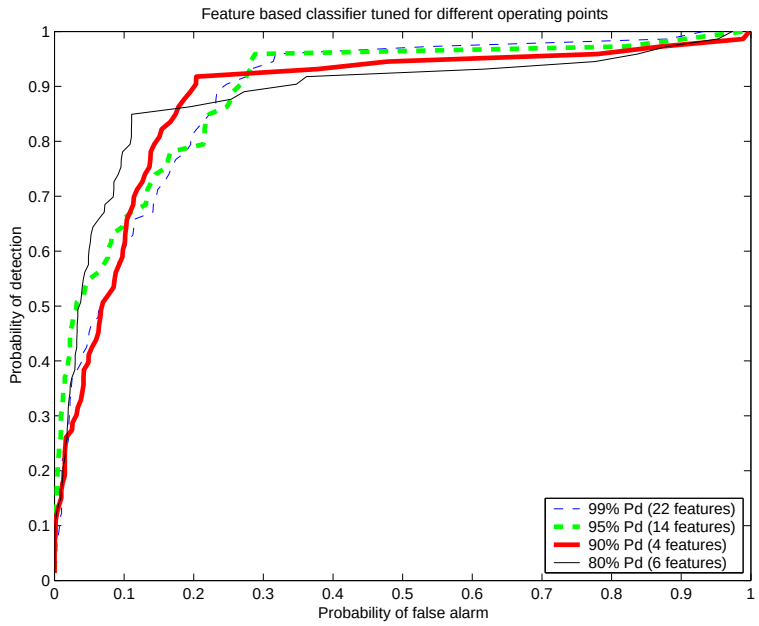


Figure 7.6: *ROC curve for subsets of features chosen at different operating points*

difference with the results obtained for the INGARA data. While some of this may be due to problems with differences in intensity scales, image resolution, different background characteristics, etc., it seems unlikely that these can account for all of the difference. A more likely source of difficulty is that over half of the targets were either camouflaged, or in scrub which allowed obscuration of the radar return. In these cases, only small portions of the vehicle may be visible, so the maximum intensity is much more likely to yield a detection than other techniques which would assume that a real target would also contain nearby background clutter pixels.

If the hypothesis in the previous paragraph were correct, it would be expected human operators would also have great difficulty locating targets. This is confirmed by the two stars in Figure 7.7 which were obtained by displaying sections of each image, and manually selecting points in the images as being either almost certainly targets, or possibly targets. The highest detection rate was only just greater than 50 percent, with most of the missed detections being in the regions where the targets were concealed in scrub. The false alarm rate for this method of detection was only just lower than that for ATA. The only reason this detection rate was so low was that there were many SAR images of the same location with different target positions, and certain background features which were thought to be targets in one image were realised to be background when they appeared in the same positions in other images. The result indicates that ATA may be the best that can be done for high detection probabilities because most of the targets are just not visible. Furthermore, when the PD is high, most of the detections which may not be useful even when the detection is correct. This is because the detection is passed to an image analyst, who may not be able to see anything useful, and will assume it is a false alarm, even when it is not.

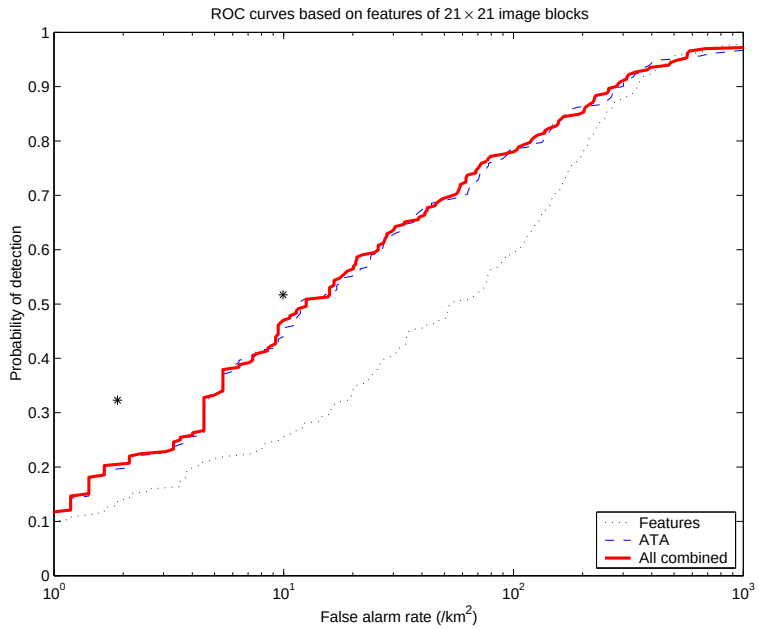


Figure 7.7: *ROC curve for a selected subset of INGARA features*

7.3 Classification

The classification of maritime targets consists of three main steps. The first is to derive a number of objective features for each of the potential targets. Due to the limited resolution of SAR systems, combined with motion blurring, there are limits to the amount of information which can be extracted from the imagery, and it is suggested that the features used for land-targets in previous reports [6, 7, 8] would be sufficient. Because classifiers suffer from the "curse of dimensionality" where the test performance becomes much worse than the training performance as the number of features increases, the next step is to reduce the number of features, and then finally use these to distinguish between ships and clutter. Since this feature selection process is usually dependent on the classifier, the classification stage is discussed first in this section. The feature selection stage is described separately in Section 7.3.4.

7.3.1 Overview of ensembles of classifiers

Recent classification techniques, such as boosting and bagging, have focused on constructing complicated decision surfaces from weighted sums of simple classifiers such as linear and quadratic discriminants and small decision trees. These ensemble based classifiers have been found to produce results comparable with the best of other techniques such as support vector machines.

One of the first ensemble classifiers studies was bagging [3], which stands for "Bootstrap AGGregatING". The term bootstrap refers to the process of generating new training sets from the original set of points by sampling with replacement. A simple discriminant

(known as the base classifier) is then applied to each of these training sets, and the bagged discriminant is then the average of all of these base classifiers. Breiman argues that the improvement in performance achieved by bagging is due to variance reduction, in that points which may not be classified correctly by one discriminant will still on average be correctly classified more often than not. Breiman's argument is further supported by the observation that robust classifiers (such as linear discriminants) which are less sensitive to the training data and so have less "variance", do not have their performance improved as much by bagging as their more sensitive counterparts (such as large decision trees).

A more sophisticated form of ensemble classifier was considered, with the introduction of the AdaBoost algorithm [11] in 1995. The idea behind this algorithm was to weight the classification algorithm toward more difficult to classify points, and to use non-uniform weighting on the classifiers to increase the effect of those that produced better results. Suppose a set of points $\mathbf{x}_n$ have known classification $y_n \in \{-1, 1\}$. Then the AdaBoost algorithm (as well as many others based on the same approach) constructs a classification function $F(\mathbf{x})$ from individual base classifiers $f_i(\mathbf{x})$, using the following generic method:

- Initialise the weighting of each of the N points to be uniform (so $w_n = 1/N$) and $F(x) = 0$.
- Repeat the following (where i is the number of times the loop has been repeated) until some convergence property has been satisfied
  - Apply the base classifier to the weighted points to give the function $f(x_i)$. If the classifier does not accept weights, a bootstrap sampling of the training set can be used instead.
  - Based on the classification of the points, determine a classifier weighting ϵ_i and update the boosted classifier using

$$F(x)' = F(x) + \alpha_i f_i(x).$$

  - Reweight the points using the base classifier and boosted classifier results.

Algorithms of the above type are referred to as leveraging algorithms. Some of these algorithms (such as AdaBoost) can be shown to minimise some upper bound on the generalisation error when the base classifiers are assumed to be weak learners (*i.e.* they almost certainly classify $(1 - \epsilon)/2$ of the training data incorrectly, for some positive ϵ). When such a bound exists, the algorithm is known as a boosting algorithm.

AdaBoost, which remains the most popular boosting algorithm, uses the weighted base classifier error $e_i = \sum_n w_n y_n f_i(x_n)/N$, with classifier weights of

$$\alpha_i = \ln\left(\frac{1 - e_i}{e_i}\right)$$

and updates the point weights using

$$w'_n = w_n \exp(\alpha y_n f_i(\mathbf{x}_n)/2).$$

The original choice of these classifier and point weight update formulae was to minimise the training error of the boosted classifier in the smallest number of steps. While this formulation explained the good training set performance of the classifier, it did not explain the high generalisation performance of the boosted classifier. This performance went against conventional wisdom that, generally, more complicated classifiers are more prone to overfitting the data and give a worse generalisation performance. With boosting however, increasing the complexity of the model by adding extra base classifiers usually seemed to improve test performance even after the training error had dropped to zero. One method for explaining this performance was the use of margins.

The margin of a training point x_i is defined to be $y_i F(\mathbf{x}_i)$. It has been shown using PAC-learning (Probably Approximately Correct) theory that by maximising the margin of a classifier, that an upper bound on the generalisation error of the classifier is minimised. This was used to derive maximum margin classifiers such as the support vector machine, which was to prove to be a very capable classifier. In a 1997 paper [25], it was shown that AdaBoost maximised some measure of the margin, and hence the PAC based results could be used to bound the generalisation error. The result has been criticised because the bound is extremely loose, and in fact since then, examples have been found where applying AdaBoost produces a classifier with worse performance than the original base classifier.

Another paper in 1997 by Breiman [4] also attempted to explain the generally remarkable performance of AdaBoosted classifiers. He approached this in a qualitative manner by referring to the concepts of bias and variance. The bias term is due to deficiencies in the classifier, where the probability of correctly classifying any given point is different from that of the Bayes' optimal solution. The variance term is due to the statistical effect of sampling, and can be removed (as in bagging) by resampling and combining classifiers. Breiman suspected that the point reweighting term in AdaBoost could be thought of as removing the bias, and that the specific form of the reweighting was unimportant. For this reason, he described a number of ARCing (Adaptively Resample and Combine) classifiers, the best of which was dubbed Arc×4. This classifier reweighted the points using the update formula

$$w'_n = w_n(1 + m(x_n)^4).$$

where $m(x_n)$ is the number of misclassifications of the nth point. The resulting classifiers were combined by unweighted voting, as in bagging. The ARCing algorithms produced performance similar to AdaBoost on the data sets tested.

One relationship between AdaBoost and Arc×4, as well as a number of other boosting algorithms (*e.g.* ConfidenceBoost and LogitBoost) is that they can be considered as gradient descent methods in discriminant function space for some loss function, as described by Mason et. al. [19]. For instance, when the loss is given by

$$\begin{aligned} L(y, F(\mathbf{x}) + \epsilon f(\mathbf{x})) &= \sum_i c(y_i(F(x_i) + \epsilon f(x_i))) \\ &\approx \sum_i c(y_i F(x_i)) + \epsilon \sum_i y_i f(x_i) c'(y_i F(x_i)) \end{aligned}$$

then, for a given ϵ, the direction of steepest descent will occur when the second summation is largest. If f belongs to the set of all possible base classifiers, then it will be the one with the lowest weighted error, where each point has been weighted by the factor $c'(y_i F(x_i))$. This means that the Arcx4 algorithm with the weighting above, would correspond to the loss function

$$L(y, F(\mathbf{x})) = \sum_i (1 - y_i F(x_i))^5$$

and that AdaBoost would similarly be a gradient descent method with the loss function

$$L(y, F(\mathbf{x})) = \sum_i \exp(-y_i F(x_i)).$$

In most optimisation problems, the direction of steepest descent is not optimal, and the direction of the global minimum of L may lie in quite a different direction. Friedman et.al. [13] showed a slightly stronger convergence result for AdaBoost, with the same loss function, in the context of additive models. This result showed that each step in AdaBoost maximised the decrease in the loss L in a greedy manner. This means that at each stage, the search direction $f(\mathbf{x})$ and the step size ϵ were jointly chosen to give the minimum possible loss at the end of each iteration.

Consider $F_n(x)$, which is the weighted sum of n classifiers. Then if the loss function is minimised at each stage of the additive model, then

$$\begin{aligned}
(\alpha_n, f_n(\mathbf{x})) &= \overset{argmin}{\alpha, f} \sum_{i=1}^{N} \exp(-y_i(F_{n-1}(\mathbf{x}_i) + \alpha f(\mathbf{x}_i))) \\
&= \overset{argmin}{\alpha, f} \sum_{i=1}^{N} w_{i,n} \exp(-y_i \alpha f(\mathbf{x_i})) \\
&= \overset{argmin}{\alpha, f} \sum_{i=1}^{N} w_{i,n} \exp(\alpha) + (\exp(-\alpha) - \exp(\alpha)) \sum_{i \in C} w_{i,n}
\end{aligned}$$

where $w_{i,n} = \exp(-y_i(F_{n-1}(\mathbf{x}_i)))$ are the point weights after $n-1$ iterations of AdaBoost and C is the set of correctly classified points. The last line is minimised when the discriminant $f_n(\mathbf{x})$ is chosen to minimise the weighted error over the training set. Substituting this weighted error $\epsilon_n = \sum_{i \in \bar{C}} w_{i,n} / \sum i w_{i,n}$ into the above equation gives

$$\begin{aligned}
\alpha_n &= \overset{argmin}{\alpha} \left(\sum_{i=1}^{N} w_{i,n} \right) (\exp(\alpha)\epsilon + \exp(-\alpha)(1 - \epsilon)) \\
&= \frac{1}{2} \ln \left(\frac{1 - \epsilon}{\epsilon} \right),
\end{aligned}$$

which is equivalent to the AdaBoost algorithm described above. As with the previous papers however, while the good training error is explained by the analysis, there is no theoretical explanation for the corresponding generalisation performance.

Empirically, AdaBoost tends to perform very well when there is little overlap between the distributions. Due to AdaBoost's margin maximisation properties however, the final classifier will tend to have a very low training error. Points from one class which stray into parts of feature space with high concentrations of the other class will still be classified correctly, despite this being of a low likelihood in the Bayes optimal case. Several algorithms have been designed attempting to correct this problem in AdaBoost. Most of them are based on loss functions which less strongly weight points that have been misclassified very strongly. For instance, DOOMII [19] uses the loss function

$$L(y, F(\mathbf{x})) = \sum_i (1 - \tanh(\lambda y_i F(x_i)))$$

for some positive constant λ.

A much more complicated boosting algorithm is called BrownBoost [12], which assumes it has a fixed "time" in which to correctly classify points. At each stage, the classifier and point reweightings are calculated by solving a differential equation which calculates how likely a point is to be correctly classified in the remaining time. Those points which are unlikely to be correctly classified are abandoned (by lowering their weights in subsequent iterations) so that a better result can be achieved on the remaining points. One of the important input parameters to this method is T, the total amount of "time" (corresponding to the total weights of the individual classifiers) available for classification. Allowing $T \to \infty$ reduces the method to the standard AdaBoost algorithm.

By examining at the point reweighting term in BrownBoost, the boosting method can be seen as a steepest descent method applied to a loss function of the form

$$L(y, F(\mathbf{x})) = \sum_i \mathrm{erf}((y_i F(x_i) - c)/t)$$

where c is some positive constant corresponding to the drift in the margin distribution due to the overfitting of the classifiers to the training data, and t is the "time" remaining for the algorithm to run. Therefore, the method starts with a rather smooth loss function which is easy to numerically minimise, with few (if any) local minima. As the "time" remaining tends to zero, the loss becomes

$$L(y, F(\mathbf{x})) = \sum_i \mathrm{sgn}(y_i F(x_i) - c)$$

which is the actual classification error of the training set. This gradual reduction of the time remaining is similar to a temperature schedule in simulated annealing optimisation problems, and has the tendency to give a global minimum in the classification error rather than just a local minimum. The total time T allocated for BrownBoost will be related to the complexity of the final output classifier. A more complicated classifier will produce a smaller error, but will be more prone to overfitting. The step size considered in the BrownBoost algorithm is obtained as the solution to a differential equation, whose origin is difficult to determine from the paper. A direct minimisation of the above loss functional may be a simpler way to obtain a similar performance.

7.3.2 Some classification results

In order to test the performance of the various classification methods described in this section, the "banana" data set has been downloaded from a web benchmark repository [27]. This data set was chosen because it has two classes, and is two dimensional which allows the resulting decision surfaces to be plotted, to display the characteristics of each of the classifiers. Because only one data set was used, the results are indicative only. To obtain a more thorough indication of the advantages and disadvantages of the methods presented, a wider range of data sets needs to be considered.

This subsection first shows a number of experiments using existing algorithms where various parameters such as the type of base classifier are altered. Then some minor implementation issues are discussed. These include sequential updating of boosting formulae as new training points become available, and stopping conditions for the boosting algorithm. The next part of the subsection presents a number of new ideas for extending boosting code. Many of these have not proved to be useful on the above data sets, but certain ideas (such as recursive boosting and the regularised discriminant based method) show a small but noticeable improvement to the standard methods.

7.3.2.1 The effect of the base classifier

Although the original AdaBoost algorithm was specifically derived for improving the performance of "weak learners", in practice it may be applied to practically any type of discriminant. The choice of base classifier can have a significant effect on the classification error of the resulting discriminant. It is usually chosen to be simple and quick because they need to be calculated a large number of times. The family of discriminants should be general enough to allow any decision surface to be modelled as a linear combination, but not too general, as this results in massive overfitting. Frequently, decision trees containing a small number of nodes (often less than five) are used as the base classifier, as it has been found that boosting large decision trees has little effect. Figure 7.8 shows the boosted classifier produced from a decision stump, linear and quadratic classifier, and a two level decision tree. It can be seen that the decision stump and the linear discriminant cannot seem to fit the data very well, and have large test errors. The slightly more flexible discriminants give significantly improved discrimination.

7.3.2.2 Sequential boosting update

Boosting algorithms produce decision surfaces which are linear combinations of their base classifiers. Typically, several thousand iterations of the base classifier are required to give useful discrimination, which leads to slow speeds when there are large numbers of points and the base classifiers are complicated. Leave one out cross-validation is a frequently used procedure for measuring the test performance of a classifier, which for some boosting problems will be impractical without some way of speeding things up. One such method would be to assume that the base classifier decision surfaces produced by the boosting algorithm remain identical, so that they don't need to be updated. The classifier weights (generally a function of the weighted training error) are easily updated however,

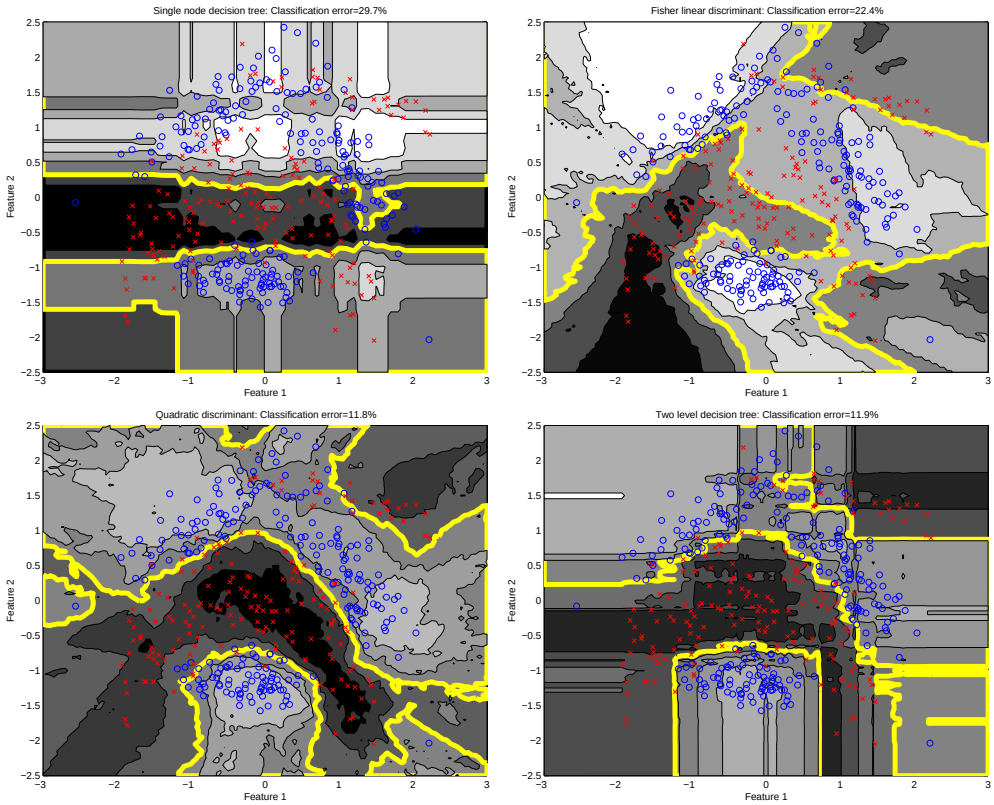


Figure 7.8: *The effect of base classifier on AdaBoost performance (number of iterations is* 1000*)*

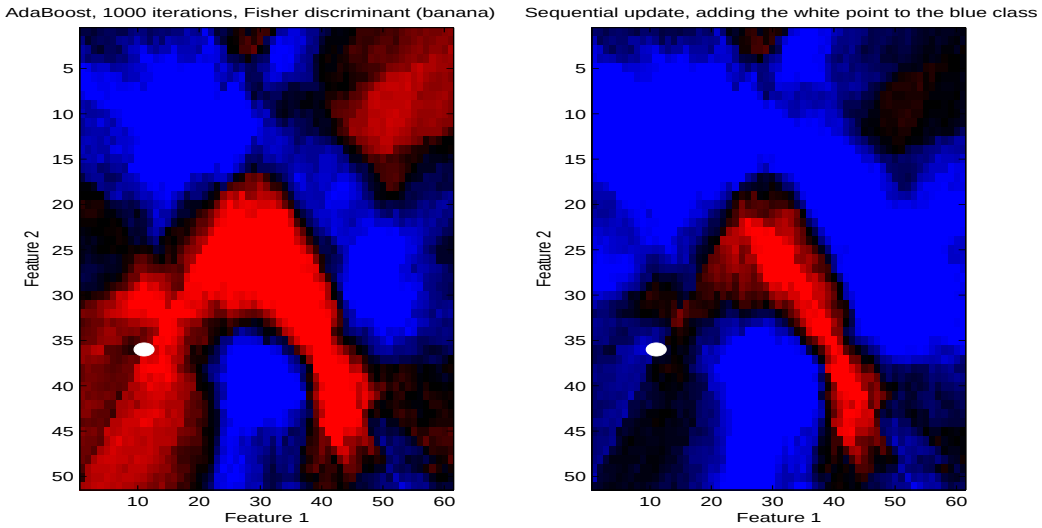


Figure 7.9: *The effect of using sequential update to add the white point to the blue class*

so the training error of the sequentially updated formula just comes down to computing a weighted sum.

Figure 7.9 gives an example of sequential update applied to the "banana" data set. In this example, the first figure shows the AdaBoost decision surface where the white point is not included in the training set. In this case, the white point is classified as belonging to the red class. If the classifier weights are modified as if the white point were added to the blue class, then the resulting decision surface is shifted, as shown in the second figure. In this case, the white point is now classified as blue, which indicates that the point is not confidently classified by the original boosting algorithm.

7.3.2.3 Boosting termination

As mentioned previously, it has been found that while boosting will tend to drive the training error to zero, frequently the test error will continue to improve. There is, however, a point at which further boosting does damage the test performance. One plausible stopping point is where the base classifier is no longer performing better than chance. This can be measured using the area under the sample based ROC curve (or AUC for Area Under the Curve) as a performance measure, and applying a one-tailed statistical hypothesis test. The null hypothesis would be that the ROC curve is generated by two identical class distributions. Unfortunately, the exact distribution of the AUC can not be determined exactly, but the mean and variance can, and can be used to provide a bound on the AUC for use in the hypothesis test. To test the null hypothesis, we make use of the following theorem.

Theorem: Suppose samples are available from two classes of points. The first class has M independent samples with weights U_i while the second class has N independent samples with weights V_j, where each of the sets of weights sums to one. Then when both classes are known to be identically distributed, the mean of the sample AUC will be one half, and the variance will be given by

$$var(AUC) = \frac{1}{12}\left(\frac{1}{M_{eq}} + \frac{1}{N_{eq}} + \frac{1}{M_{eq}N_{eq}}\right) \tag{1}$$

where M_{eq} and N_{eq} are the equivalent number of samples for the two classes, given by

$$M_{eq} = 1/\sum_i U_i^2 \qquad\qquad N_{eq} = 1/\sum_j V_j^2.$$

Proof: The ROC curve consists of a set of M horizontal and N vertical line segments. The order in which these lines occur will depends on the particular set of samples. Because the two classes are identically distributed, the probability that any given line will occur will be equal. Now based on Figure 7.10, the mean of the AUC for the given weights can be written as

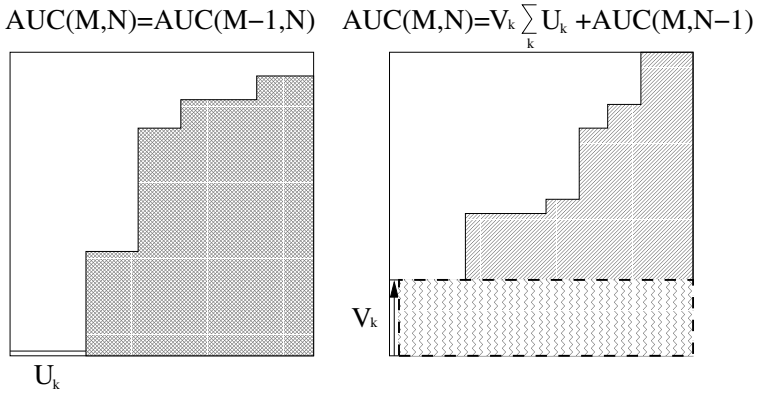


Figure 7.10: *Geometrical depiction of difference equation (2)*

$$\begin{aligned} A\bar{U}C(U_{1\ldots M}, V_{1\ldots N}) &= \sum_k \frac{1}{M+N} A\bar{U}C(U_{i\neq k}, V_{1\ldots N}) \\ &\quad + \sum_k \frac{1}{M+N}\left(\sum_i U_i V_k + A\bar{U}C(U_{1\ldots M}, V_{j\neq k})\right) \end{aligned} \tag{2}$$

The following proposition $P(M,N)$ is put: that the solution to this difference equation is $A\bar{U}C(U_{1\ldots M}, V_{1\ldots N}) = \sum U_i \sum_j V_j/2$. Now obviously $P(M,0)$ and $P(0,N)$ are true. Assuming $P(M-1,N)$ and $P(M,N-1)$ are true gives

$$\begin{aligned} RHS &= \frac{1}{M+N}\left(\sum_k \sum_{i\neq k} U_i \sum_j V_j/2 + \sum_i U_i \sum_j V_j + \sum_i U_i \sum_k \sum_{j\neq k} V_j/2\right) \\ &= \frac{1}{M+N}\left(\frac{M-1}{2} + \frac{N-1}{2} + 1\right)\sum_i U_i \sum_j V_j \\ &= \frac{1}{2}\sum_i U_i \sum_j V_j \end{aligned}$$

which is $P(M,N)$. Therefore, by induction, the proposition is true and because the weights sum to one, the mean AUC is one half.

Similarly, the variance of the AUC can be described by the difference equation

$$\begin{aligned} var(AUC(U_{1\ldots M}, V_{1\ldots N})) &= A\bar{U}C^2(U_{1\ldots M}, V_{1\ldots N}) - \frac{1}{4}\left(\sum_i U_i \sum_j V_j\right)^2 \\ &= \sum_k \frac{1}{M+N} A\bar{U}C^2(U_{i\neq k}, V_{1\ldots N}) - \frac{1}{4}\left(\sum_i U_i \sum_j V_j\right)^2 \end{aligned}$$

$$
\begin{aligned}
&+\sum_k \frac{1}{M+N} E\left(\sum_i U_i V_k + AUC(U_{1\ldots M}, V_{j\neq k})\right)^2 \\
= \;& \frac{1}{M+N}\sum_k var(AUC(U_{i\neq k}, V_{1\ldots N})) - \frac{1}{4}\left(\sum_i U_i \sum_j V_j\right)^2 \\
&+\frac{1}{4(M+N)}\sum_k \left(\sum_{i\neq k} U_i \sum_j V_j\right)^2 + \frac{1}{M+N}\sum_k (\sum_i U_i)^2 V_k^2 \\
&+\frac{1}{M+N}\left((\sum_i U_i)^2 \sum_k V_k \sum_{j\neq k} V_j + \sum_k A\bar{U}C^2(U_{1\ldots M}, V_{j\neq k})\right) \\
= \;& \frac{1}{M+N}\left(\sum_k var(AUC(U_{i\neq k}, V_{1\ldots N}) + \sum_k var(AUC(U_{1\ldots M}, V_{j\neq k}))\right) \\
&+\frac{1}{4(M+N)}\left(\sum_k \left(\sum_{i\neq k} U_i\right)^2 \left(\sum_j V_j\right)^2 + \left(\sum_i U_i\right)^2 \sum_k \left(\sum_{j\neq k} V_j\right)^2\right) \\
&+\left(\frac{1}{M+N} - \frac{1}{4}\right)\left(\sum_i U_i\right)^2 \left(\sum_j V_j\right)^2 \\
= \;& \frac{1}{M+N}\left(\sum_k var(AUC(U_{i\neq k}, V_{1\ldots N}) + \sum_k var(AUC(U_{1\ldots M}, V_{j\neq k}))\right) \\
&+\frac{1}{4(M+N)}\left(\sum_k U_k^2 \left(\sum_j V_j\right)^2 + \left(\sum_i U_i\right)^2 \sum_k V_k^2\right)
\end{aligned}
$$

Now if we define $U1 = \sum_i U_i$, $U2 = \sum_i U_i^2$, $V1 = \sum_j V_j$ and $V2 = \sum_j V_j^2$, then we make the following proposition $P'(M,N)$: that the solution to the above difference equation is $var(AUC(U_{1\ldots M}, V_{1\ldots N})) = (U2V1^2 + V2U1^2 + U2V2)/12$. This will obviously be true for $M = 0$ or $N = 0$. Assuming $P'(M-1,N)$ and $P'(M,N-1)$ are true and substituting into the right hand side of the above equation gives

$$
\begin{aligned}
&\frac{1}{12(M+N)}\left\{\sum_k \left[(U2 - U_k^2)(V1^2 + V2) + (U1 - U_k)^2 V2\right]\right. \\
&\left.+\sum_k \left[(V2 - V_k^2)(U1^2 + U2) + (V1 - V_k)^2 U2\right] + 3(U2V1^2 + V2U1^2)\right\} \\
= \;& \frac{1}{12(M+N)}\left[(M-1)U2(V1^2 + V2) - U1^2 V2 + U2V2\right. \\
&\left.+(N-1)V2(U1^2 + U2) - V1^2 U2 + U2V2 + 3(U2V1^2 + V2U1^2)\right] \\
= \;& \frac{1}{12}(U2V1^2 + V2U1^2 + U2V2) \\
= \;& LHS
\end{aligned}
$$

which is proposition $P'(M, N)$. Therefore, by induction the proposition is true in general. It is easily seen that since the weights sum to one that the theorem must be correct.

QED

Some preliminary data obtained using the banana data set indicates that the AUC test statistic given by $(AUC - 1/2)/\sqrt{var(AUC)}$ varies more or less randomly from one boosting iteration to the next, and there is no clear downwards trend. Although this indicates that a simple threshold on the base classifier is not likely to prove useful, there are other ways in which the above theoretical result may prove useful in determining a stopping point for boosting. Due to a lack of time however, these have not yet been tested.

7.3.3 New methods

In this subsection, some new methods for classifying training data are discussed. While many of these performed poorly compared to the methods described previously, incorporation of at some of the ideas behind the methods into other classifier schemes may prove useful.

7.3.3.1 Combining supervised and unsupervised learning

Due to the restricted number of points available in many maritime classification problems, it is useful to be able to use as many of these points as possible in training the classifier. An obvious approach is to evaluate classifier performance using a leave-one-out method. In maritime imaging radar applications though, the difficulty is not necessarily a lack of points, but a lack of usable points due to the unavailability of ground-truth on most of the candidate detections within the image scene. This problem might be partially overcome by fusing the output of supervised and unsupervised learning.

The first algorithm that was implemented worked as follows:

- **Supervised classification:** A classifier is applied to the data with assigned labels (initially just the training data). The classifier produces a weight p_i related to the likelihood of a point i belonging to a particular class (say class 1). This weight is calculated for all of the points in the test and cross-validation set with unknown class.
- **Unsupervised classification:** The points from the cross-validation and test sets are then clustered with reference to their weights p_i. This can be done using an Expectation Maximisation (EM) algorithm to model the distribution by a mixture of two Gaussians. The mixture having the larger average p_i may then be labelled as either class 1 or class 2, depending on the likelihood ratio of the two Gaussians at that point.
- **Iteration:** The above two steps are iterated until none of the points change their label, and the resulting classifier is used to assess the performance.

The above algorithm was tested on a data set containing a few hundred data points in total, and ten thousand feature vectors. The resulting classification was no better than that of the original classifier. This is likely because any chosen labelling of the unknown points could result in a perfect classification on the next iteration due to the large feature dimensionality. In a maritime surveillance scenario, where the number of the non-target class available is much larger, the method may work better, although this has not yet been tested.

7.3.3.2 Outlier detection

Most boosting methods, such as AdaBoost, attempt to classify all of the training points correctly. The optimal Neymann-Pearson distribution however will, in many cases, produce a non-zero training error due to overlap between the class distributions. Qualitatively, it has been found that the more overlap in the data (either due to poor feature separation or mislabelled training points), the less optimal the resulting boosted classifier will be. One method for reducing this overlap would be to remove the points which should not be classified correctly (or anomalies) prior to boosting. Obviously, if all of these points could be removed accurately, there would be no need for a boosting classifier. The idea is that if some classification can be applied prior to boosting to remove problem points, then the boosted classifier might prove even more accurate than the original classifier.

One of the most commonly used non-parametric classifiers in the literature is the k nearest neighbour classifier. In this case, a test point is classified as the highest frequency class from the closest k points of the training set. Such a classifier could be used to determine some of the training points to ignore prior to boosting. A typical problem in this type of classifier however, is how to determine the best value of k. This problem has been approached by Ghosh et. al. [21] by considering the Bayesian evidence (described as the "Bayesian strength function" in [21]) for making a classification prediction for each value of k.

Suppose a given test point has n_1 neighbouring training points from the first class, and $n_2 = k - n_1$ neighbours from the second class. Further suppose that the probability of the test point being generated from the first class is p (and from the second class as $1 - p$). Then the evidence that the test point should be classified as the first class (*i.e.* that $p > N_1/(N_1 + N_2) = a$, where N_1 and N_2 are the number of training examples from each class) is given by Baye's theorem as

$$S = P(p > a|n_1, n_2) = \frac{\int_{p>a} P(n_1, n_2|p)\pi(p)}{\int P(n_1, n_2|p)\pi(p)} = \frac{\int_a^1 p^{n_1}(1-p)^{n_2}dp}{\int_0^1 p^{n_1}(1-p)^{n_2}dp}.$$

where $\pi(p)$ is the prior distribution of probabilities which, for lack of a better alternative, has been chosen as the uniform distribution (it has been reported that S is quite insensitive to the choice of π). In [21], the above integrals were estimated for each of the test points for each value of k using Monte-Carlo methods. Here, an exact iterative solution is provided to evaluate the integrals more quickly. Define

$$I(a, n_1, n_2) = \int_a^1 p^{n_1}(1-p)^{n_2}dp. \tag{3}$$

Let $u = p^{n_1}$, so $du = n_1 p^{n_1-1} dp$ and $dv = (1-p)^{n_2} dp$ so $v = -(1-p)^{n_2+1}/(n_2+1)$. Then integrating by parts gives

$$\begin{aligned} I(a, n_1, n_2) &= \left[p^{n_1} \frac{-(1-p)^{n_2+1}}{n_2+1} \right]_a^1 + \int_a^1 \frac{(1-p)_{n_2+1}}{n_2+1} n_1 p^{n_1-1} dp \\ &= \frac{a^{n_1}(1-a)^{n_2+1}}{n_2+1} + \int_a^1 \frac{(1-p)^{n_2}}{n_2+1} n_1 p^{n_1-1} dp - \int_a^1 \frac{(1-p)^{n_2}}{n_2+1} n_1 p^{n_1} dp \\ &= \frac{a^{n_1}(1-a)^{n_2+1}}{n_2+1} + \frac{n_1}{n_2+1} I(a, n_1-1, n_2) - \frac{n_1}{n_2+1} I(a, n_1, n_2) \end{aligned}$$

which after rearrangement gives

$$(n_1 + n_2 + 1) I(a, n_1, n_2) = a^{n_1}(1-a)^{n_2+1} + n_1 I(a, n_1 - 1, n_2) \tag{4}$$

Similarly, setting $u = (1-p)^{n_2}$ so $du = -n_2(1-p)^{n_2-1}$ and $dv = p^{n_1} dp$ so $v = p^{n_1+1}/(n_1+1)$, and using this to integrate equation (3) by parts for $n_2 \geq 1$ gives

$$\begin{aligned} I(a, n_1, n_2) &= \left[(1-p)^{n_2} \frac{p^{n_1+1}}{n_1+1} \right]_a^1 + \int_a^1 \frac{p^{n_1+1}}{n_1+1} n_2 (1-p)^{n_2-1} dp \\ &= -(1-a)^{n_2} \frac{a^{n_1}+1}{n_1+1} + \frac{n_2}{n_1+1} \int_a^1 (-(1-p)+1) p^{n_1} (1-p)^{n_2-1} dp \\ &= -(1-a)^{n_2} \frac{a^{n_1}+1}{n_1+1} + \frac{n_2}{n_1+1} \int_a^1 p^{n_1} (1-p)^{n_2-1} dp \\ &\quad - \frac{n_2}{n_1+1} \int_a^1 p^{n_1} (1-p)^{n_2} dp \\ &= -(1-a)^{n_2} \frac{a^{n_1}+1}{n_1+1} + \frac{n_2}{n_1+1} I(a, n_1, n_2-1) - \frac{n_2}{n_1+1} I(a, n_1, n_2) \end{aligned}$$

and hence

$$(n_1 + n_2 + 1) I(a, n_1, n_2) = -(1-a)^{n_2} a^{n_1+1} + n_2 I(a, n_1, n_2 - 1). \tag{5}$$

Now to calculate the evidence $S = I(a, n_1, n_2)/I(0, n_1, n_2)$, for a given test point, the number of nearest neighbours from the test set (k) starts at zero, and is repeatedly incremented. If the added point is from the first class, then equation (4) can be used to update the integrals. Otherwise equation (5) may be used to evaluate the evidence for each value of k chosen.

The above model for the Bayesian evidence has been implemented, but its use in anomaly detection has not yet been fully tested. This is because there is some difficulty in determining the optimum value for the number of nearest neighbours for each point. The original intention was to use the value of k for which the evidence was greatest towards one class or the other. Unfortunately, this usually occurred for large values of k for which the implicit assumptions that the classes have constant density over the neighbourhood become incorrect. The evidence function therefore also needs to include a factor which

reduces the evidence of both classes when the density (or more specifically, the relative density) of the classes change. This can be measured by observation of the class labels of the set of neighbourhood points, ordered by distance from the test point. In the case where the relative densities are equal, the class labels would be distributed by a Markov model, with transition probabilities related to the ratio of the densities. If the densities were to change, the Markov model would fit less well. At the moment, this model has not been tested, but is an area where future work might prove useful.

7.3.3.3 Repeated boosting

Boosting methods are procedures that can be applied to any classifier, including one that has already been boosted. In the case where the boosting also sums the confidence of prediction of the base classifiers, the result of boosting a boosted classifier will also be a linear sum of that same base classifier. Therefore, if a particular boosting method is optimal across all classifiers, then it should converge to precisely the same decision surface. For AdaBoost, this does not seem to be true numerically.

Some numerical tests have been applied to 100 different instances of the 'banana' data set using multiple levels of AdaBoost. Using 1024 iterations of standard AdaBoost to a 2 node decision tree gave a mean classification error of 14.4 ± 0.08. To test the effect of repeated boosting without changing the total number of base classifiers, 4 iterations of AdaBoost were applied to a 256 iteration AdaBoosted decision tree, which gave a mean error of 14.0 ± 0.07. Although this is an improvement, it is not very large. An example of the resulting classifier is shown in Figure 7.11.

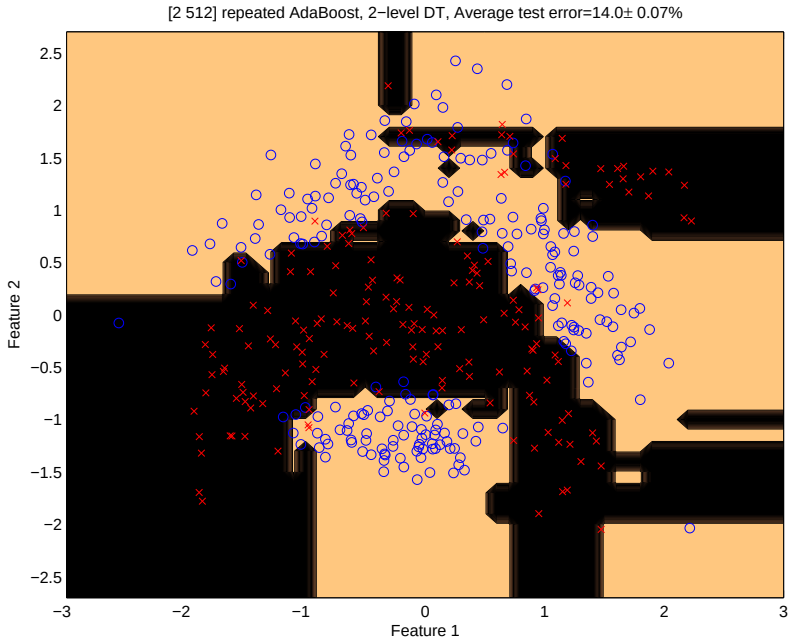


Figure 7.11: *Boosting a boosted decision stump classifier*

There were a number of ways in which the number of iterations in each level of repeated boosting can be chosen to give the same overall number of base classifiers. It was found that at least one of the levels needed a large number of iterations for there to be any advantage to repeated boosting. For instance, when both levels used 32 iterations, the resulting classification error was 14.9 ± 0.09, which was higher than just using the one level of boosting.

After adding a second level of boosting, it is natural to consider adding a third level. Keeping in mind the previous empirical observation that at least one of the levels seems to need a large number of boosting iterations, two examples were chosen where the levels had $\{2, 2, 256\}$ and $\{4, 4, 64\}$ iterations respectively. The classification errors were measured to be 13.8 ± 0.08 and 15.2 ± 0.1. The first result indicates a minor improvement over both the single and double level boosting results. The second result however did not use any level with a large number of iterations so, as in the two level example, the classification was worse.

The problem of multiple levels of boosting can be considered mathematically. For the example in Figure 7.12, the input to some boosting (or leveraging) algorithm is: N, the total weight of the base classifiers in the current classifier, c_i the current estimate of the class for each point $\mathbf{x}_i$, and w_i the current weight for each training point. The boosting process then consists of a large (perhaps infinite) number of iterations of the base classifier, which then modifies the values of N, c_i and w_i in some way, depending on C (the output from the base classifier) and the inputs. Two successive base classifiers can then be combined into a single equivalent base classifier, and boosting applied to that.

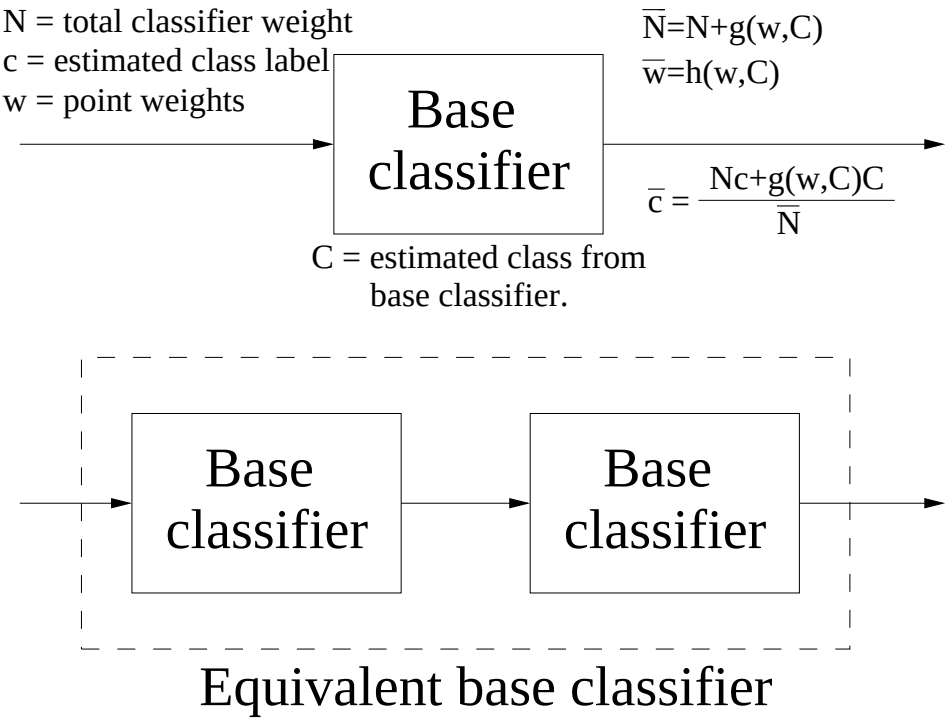


Figure 7.12: *Example of boosting a boosted classifier*

Any optimal boosting method would be expected to give rise to the same combination of base classifiers, regardless of whether the original or the equivalent base classifier was used. This property of a boosting method will be referred to as consistency.

Only a small amount of time has been spent on formulating conditions on the functions g and h which produce consistency of a boosting algorithm, for this report. So far, no useful results have been found, although it is an area that might benefit from further study. It is fairly obvious to see however that bagging is consistent. In this case, the function g is constant and h is a random resampling function. Consistency does not seem to guarantee optimality since, as commented on previously, bagging does not generally perform as well as other ensemble methods.

7.3.3.4 Choice of classifier weights

Although the classifier weight formulae obtained by AdaBoost and other boosting algorithms can be thought of as minimising some loss function on the training error, it is also worthwhile comparing the results to other weighting methods. One such method assumes that each of the base classifiers is independent, and correctly classifies individual points independently with a probability p_i for the ith classifier. The boosted classifier will therefore be $\sum_i w_i X_i$ where X_i is $+1$ with probability p_i and -1 with probability $1-p_i$. It is now required to find the classifier weights w_i which give the largest training discrimination. This can be found by maximising the Fisher separation

$$\begin{aligned}
\text{Separation} &= \frac{E\left(\sum_i w_i X_i\right)^2}{E\left(\left(\sum_i w_i X_i - \sum_i w_i(2p_i-1)\right)^2\right)} \\
&= \frac{\left(\sum_i w_i(2p_i-1)\right)^2}{\sum_i w_i^2 E\left((X_i-(2p_i-1))^2\right)} \\
&= \frac{\left(\sum_i w_i(2p_i-1)\right)^2}{\sum_i 4w_i^2(1-p_i)^2 p_i + p_i^2(1-p_i)} \\
&= \frac{\left(\sum_i w_i(2p_i-1)\right)^2}{\sum_i 2w_i^2 p_i(1-p_i)}
\end{aligned}$$

Setting the derivative with respect to the jth weight equal to zero gives

$$\frac{\partial}{\partial w_j} = 2\frac{(2p_j-1)\sum_i w_i(2p_i-1)}{\sum_i 4w_i^2 p_i(1-p_i)} - \frac{\left(\sum_i w_i(2p_i-1)\right)^2 2w_j p_j(1-p_j)}{4\left(\sum_i w_i^2 p_i(1-p_i)\right)^2} = 0$$

Rearranging this gives

$$(2p_j-1)\sum_i w_i^2 p_i(1-p_i) = w_j p_j(1-p_j)\sum_i w_i(2p_i-1)$$

The two sums will produce some arbitrary constant, so the optimal classifier weights will be given by

$$w_j = \frac{(2p_j - 1)K}{p_j(1 - p_j)} \tag{6}$$

As expected, when the probability of a correct classification is 0.5 (*i.e.* the classification is random), then the corresponding classification weight is zero. This result is similar to a matched filter for detecting signals in Gaussian noise, but here the measurement error is binomial. This theoretically justified result however gives extremely poor results (in fact, almost random) results for real data sets. This is probably due to the fact that the classification of neighbouring points (*i.e.* belonging to the same cluster) by classifiers will in general be strongly correlated, and this was not taken into account in the previous analysis.

An alternative to choosing the weights as the classifiers are calculated is to wait until all of the classifiers have been produced, and then using all of the available data, decide on the classifier weights. This is effectively a linear discriminant problem in classifier space, as compared to the original feature space. Since the number of classifiers will generally exceed the number of training examples, a Fisher linear discriminant will cause overfitting, as is shown in the first diagram of Figure 7.13 for 1000 iterations of quadratic boosting. Here, the training error is effectively zero, but the classification error is significantly worse than that obtained originally in Figure 7.8. In the previous report [5], where similar problems were encountered with respect to the calculation of prescreening templates, it was found that using a gradient descent type method to maximise the separation, and stopping before convergence generally resulted in an improved template. This appears to also hold true in the second diagram of Figure 7.13, where the classifier weights were initialise to the output of AdaBoost, and the class separation was maximised using the MATLAB Nelder-Mead based optimisation package for 10,000 function evaluations (this number was chosen arbitrarily). The test error of the resulting decision surface appears slightly less than that obtained using standard AdaBoost.

One of the techniques decided upon for the prescreener templates was regularised discriminant analysis, which was the theoretical equivalent of adding white noise to the

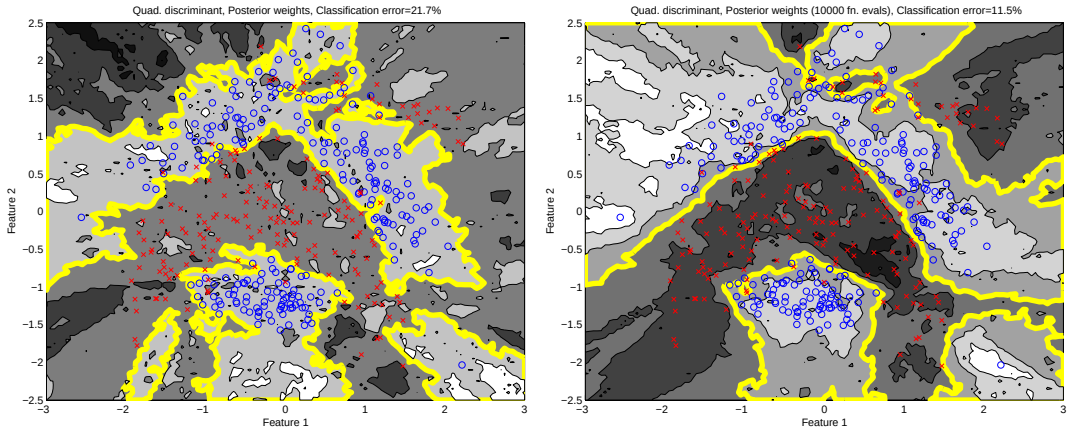


Figure 7.13: *Boosting a quadratic discriminant with posterior calculation of classifier weights*

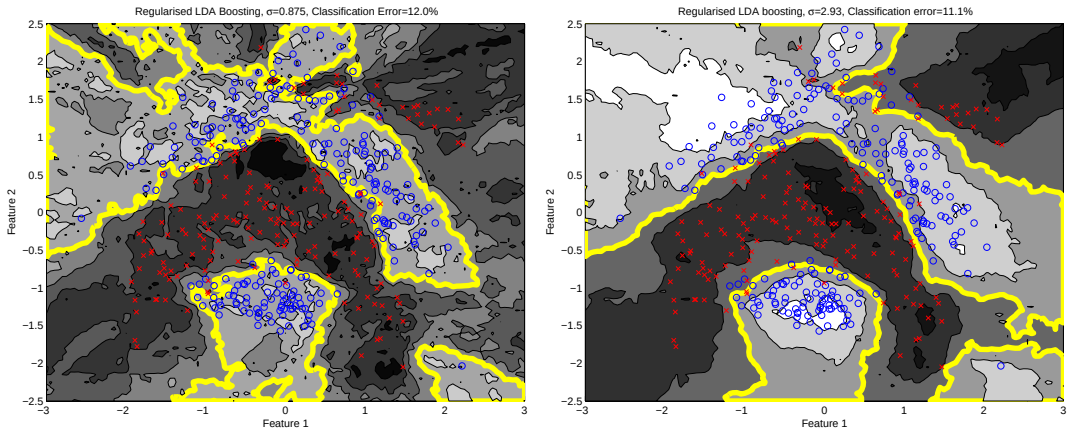


Figure 7.14: *Boosting using regularised linear discriminant analysis*

images to create a larger training set. In this case, a labelling error (corresponding to a regularisation term) is added to each of the classifiers to notionally produce some extra training set points. In the left diagram of Figure 7.14, the same labelling error (with variance σ^2) has been applied equally to all points, and has been increased slowly until the training error is no longer zero. This decision surface shows signs of overfitting, which is reduced for the second example where σ was further increased until the training error became 5 percent. The second case shows that the addition of the labelling error regularisation improves the test error in this example.

7.3.3.5 Recursive boosting

Recursive boosting (as distinct from repeated boosting discussed in 7.3.3.3) relies on an idea that was the basis for the successful recursive Fisher discriminant [7]. It uses the fact that parts of the class distributions which are far from the decision surface may be modified, or even completely excised, without the maximum likelihood decision threshold. For classifiers such as the Fisher linear discriminant, which have a strong dependence on far out points, the removal of these irrelevant points can have an enormous improvement on the classifier performance. Boosting is slightly different, in that points that are consistently classified well (and hence are far from the decision surface in some sense) will automatically be given reduced weights, so their removal may not significantly affect classification. In methods like AdaBoost though, points which are infrequently classified correctly and so would lie on the other side of the decision surface, are given much higher weights. Methods such as BrownBoost attempt to adjust for this, but here the same thing is attempted using recursion.

The recursive boosting works by first calculating the boosted classifier for the original training set. The point that is worst classified is then removed from the training set, and the boosting performed again. The process is continued until the training error of the classifier is zero. Obviously, this recursive process does not work at all in cases where the boosted training error is already zero, so for the greatest effect, either the base classifier should be made very inflexible (for instance a decision stump, which is a decision tree

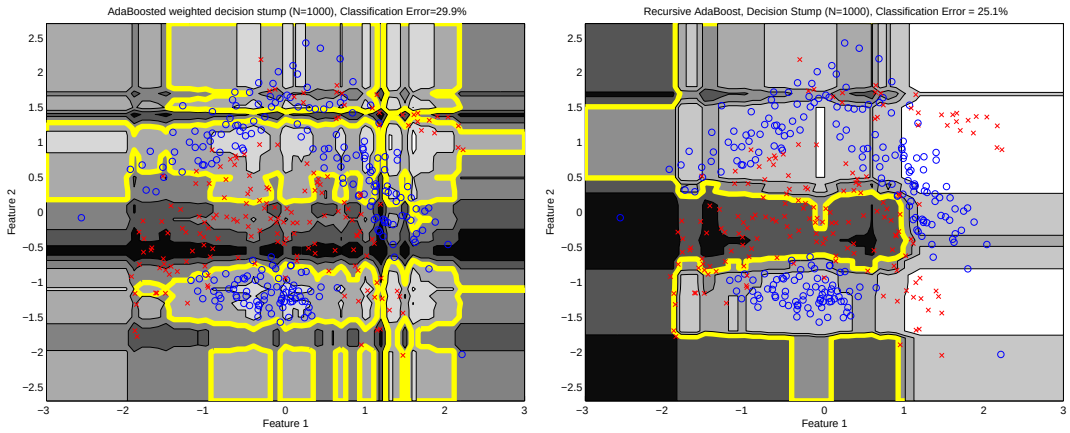


Figure 7.15: *The effect of recursion on a boosted decision stump classifier*

with one node), or the number of iterations in the boosting procedure should be made very small. Figure 7.15 shows the effect of recursive boosting on a decision stump classifier, where the boosting step is based on point weighting rather than adaptive resampling.

7.3.4 Feature reduction

One standard technique for low level classification is to construct a number of feature measures for each instance in a training set. In practice, only a small subset of these are of significant benefit to the classifier performance, so it is often prudent (both from the point of view of classifier error and computational cost) to perform some sort of feature reduction. In a previous report on target detection in SAR imagery [7], a few hundred features were reduced to about 10 using a simple forwards/backwards selection scheme. In this method, features were added or deleted from a selected feature list by their ability to reduce the training error of a classifier. Due to the relative ease of collecting SAR data for land targets, the number of targets available in the training set was still significantly larger than the number of dimensions. For maritime imagery, the reverse may be true, and so it is useful to re-examine methods for feature selection and classification in the case where the feature space dimensionality exceeds the number of points.

Feature selection (rather than reduction) is very useful in situations where the calculation of each additional feature for use in the classifier can have a large cost involved. It may however be counterproductive in cases where the individual features are individually useless for classification (such as the individual pixels intensities from an image), but may contain a great deal of information when considered together.

Feature selection has some similarity to the classifier ensemble methods described in the previous subsection. The ensemble methods choose a finite linear combination of functions f_i belonging to the space of all possible classifier decision surfaces, so this may be considered to be a dimension reduction in feature space. The most popular boosting algorithm, AdaBoost, chooses its weights in such a way that the reweighted training error (or pseudo-error) of the current estimate of the best classifier is exactly fifty percent. This

means that the next classifier chosen based on this reweighted set, in some sense, contains orthogonal information to the existing classifiers. The same approach can be applied to feature selection, in the following algorithm

- **Step 0:** Initialise the feature set $F = \phi$ and the point weighting $w_i = 1$.
- **Step 1:** Find the single best feature for discriminating the existing data. For speed purposes, a simple method such as the Fisher separation metric (difference in class means divided by the sum of the variances) can be used. Alternatively, the Kolmogorov-Smirnov or similar statistic can be used to compare the sample cumulative distribution functions. If the best feature already belongs to F then stop, otherwise add the feature to F.
- **Step 2:** Using the features F, use AdaBoost or a similar leveraging algorithm to classify the training data. Then use the AdaBoost point reweighting distribution on the margin distributions on the output to update the point weights w_i for the feature selection. Then return to **Step 1**.

The usual implementation of a forwards selection algorithm chooses a new feature by testing the performance of the classifier both with and without each of the prospective features. The above algorithm however requires one boosted classifier to be calculated for the existing features, and then a simple one dimensional discriminant to be applied to each of the remaining features.

It was originally planned that feature selection would be examined in more detail, but due to lack of time, even this first algorithm was not thoroughly tested. A quick test was implemented using a data set downloaded from the NIPS conference web site. This consisted of about 100 points from each of two classes, with each point having 1000 features associated with it. Approximately 90 percent of these were white noise, although information concerning which features were potentially useful was not provided. The above algorithm reduced the dimensionality to about 30, so it seems many useful features were discarded. Unfortunately it is not possible to determine anything else about the performance of the feature selection on this data set.

7.4 Conclusion

This report basically consists of two separate sections. The first concerned methods for extracting polarimetric and invariant transform based feature templates from imagery for use in low level classification. The second concerned different methods for fusing the information from various classifiers to improve the discrimination between targets and backgrounds in feature space. The emphasis was on boosting and other ensemble classifiers and their relation to feature selection.

The section concerning feature extraction provided a number of results for the RAAT data set containing 0.3m spotlight SAR imagery of a number of targets. For all of these results, only a very small performance improvement could be achieved over standard ATA for a 90 percent detection probability. After a simulation using human vision as the

target detection tool (similar to, but less extensive than the human vision analyst trials described in Ewing, Redding and Kettler [10]), it was concluded that most of the targets were almost completely obscured, and it was unlikely that a better detection algorithm could be found than the ATA prescreener. Unfortunately, just because an algorithm has a high detection probability doesn't mean that even correct detections are useful! This is because the detection is sent to a human analyst, and if the analyst thinks it is a false alarm, it doesn't matter whether it is a correct detection or not. A more useful measure to use for detection performance in ADSS is the probability of detecting a target that a human analyst would have noticed, which for this data set is hugely different from the detection probability actually used. For this reason, all of the results generated in Section 7.2 are not of any particular use. The performance of the features would need to be performed on either a different data set, or only the subset of targets that are actually visible, to produce useful data. In either of these cases, it is expected that the performance of the feature based methods would be significantly better than ATA.

The second part of the report concerns classification algorithms, specifically ensemble classifiers. A number of important papers concerning boosting and bagging were summarised, some algorithms were tested, and some new ideas were proposed for improving the performance of ensemble methods. Amongst the more promising ideas were recursive boosting (where training points that could not be classified correctly were removed recursively) and regularised discriminant based boosting, where the classifier weights were calculated at the end of a round of boosting using linear discriminant analysis. Both of these methods provided decision surfaces that were smoother and gave better generalisation than standard boosting for the particular data set tested. A more varied collection of data sets needs to be tested before their performance can be accurately assessed however.

The classification section also provides the basis for a number of possible directions for further research. For instance, the theorem deriving the statistics for the area under the ROC curve could be of use in deciding when to terminate boosting iterations, or as a statistical measure of the usefulness of individual features. Similarly, the work relating to Bayesian evidence of nearest neighbour classifiers may be modified, as suggested in Section 7.3.3.2, to estimate hidden Markov model parameters, to aid in outlier detection prior to (or even instead of) boosting. Also, further work could be done to examine the requirements for consistency of a boosting algorithm, as described in Section 7.3.3.3.

In summary, the report has described new but generic methods for feature extraction and classification. While it is expected that some of these methods would be of use in maritime target detection, this cannot be determined without applying them to real maritime data. Due to the lack of available ground-truthed data, this was not possible for this report.

References

1. M.Abramowitz and I.Stegun, "Handbook of Mathematical Functions," New York: Dover, 1970.

2. G.Blucher, D.Blacknell, N.Redding and D.Vagg, "Prescreening algorithm assessment

within the Analysts' Detection Support System, " Proceedings of RADAR 2003, Adelaide, 2003.

3. L.Breiman, "Bagging predictors," Machine Learning, Vol.26, pp.123–140, 1996.

4. L.Breiman, "Arcing classifiers (with discussion)," Annals of Statistics, Vol.26, pp.1493–1517, 1998.

5. T.Cooke, "First report on SAR image analysis in maritime contexts: Prescreening," CSSIP-CR-20/03, December 2003.

6. T.Cooke, "First report on features for target/background classification," CSSIP CR-9/99, April 1999.

7. T.Cooke, "Second report on features for target/background classification," CSSIP CR-26/99, November 1999.

8. T.Cooke, "Third report on features for target/background classification," CSSIP CR-5/00, May 2000.

9. T.Cooke, "Discriminant based classification," CSSIP CR-25/99, October 1999.

10. G.Ewing, N.Redding and D.Kettler, "Image Analyst's Performance in Search Mode Target Detection for Broad Area Surveillance," DSTO-RR-0110, 1997.

11. Y.Freund and R.Schapire, "Experiments with a new boosting algorithm," Machine Learning: Proceedings of the Thirteenth International Conference, Morgan Kauffman, San Francisco, pp.148–156, 1996.

12. Y.Freund, "An adaptive version of the boost by majority algorithm," Machine Learning, Vol.43, No.3, pp.293–318 June 2001.

13. J.Friedman, T.Hastie and R.Tibshirani, "Additive logistic regression: a statistical view of boosting (with discussion)," Annals of statistics, Vol.28, pp.337-367, 2000.

14. Gonzalez and Woods, "Digital Image Processing," Addison-Wesley, 1992.

15. G. Guirong, G. Xiuhuang, Y. Wenxian and H. Songhua, "An effective method for ship target recognition," SPIE, 1995, pp.606-609.

16. M. Hu, "Visual pattern recognition by moment invariants," IRE Transactions on Information Theory, IT-8, 1962, pp.179-182.

17. R.Jones, T.Cooke, N.Redding, "Implementation of the Radon transform using non-equispaced discrete Fourier transforms," DSTO-TR-1576, April 2004.

18. Z.Lei, D.Keren and D.Cooper, "Computationally fast Bayesian recognition of complex objects based on mutual algebraic invariants," ICIP 1995, Vol. II, pp.635–638.

19. L.Mason, P.Bartlett, J.Baxter and M.Frean, "Boosting algorithms and gradient descent," Advances in Neural Information Processing Systems 12, edited by S.Solla, T.Leen & K. Müller, pp.512–518, 1999.

20. F.Mokhtarian and A.Mackworth, "A theory of multiscale, curvature-based representation for planar curves," IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.14, No.8, pp.789–805, August 1992.

21. A.K.Ghosh, P.Chaudhuri and C.A.Murthy, "On visualization and aggregation of nearest neighbour classifiers," accepted by Journal of Multivariate Analysis.

22. L.Novak and M.Burl, "Optimal speckle reduction in polarimetric SAR imagery," IEEE Transactions on Aerospace and Electronic Systems, Vol.26, No.2, pp.293-305, March 1990.

23. E.M.Persoon and K.Fu, "Shape discrimination using Fourier descriptors," Proceedings of IJCPR, pp.126–130, 1974.

24. "The Transforms and Applications Handbook," ed. A.Poularikas, CRC-Press, 2000.

25. R.Schapire, Y.Freund, P.Bartlett and W.Lee, "Boosting the margin: a new explanation for the effectiveness of voting methods," Annals of Statistics, Vol.26, No.5, pp.1651–1686, 1998.

26. C.C.Wackerman, K.S.Friedman, W.G.Pichel, P.Clemente-Colón and X.Li, "Automatic detection of ships in RADARSAT-1 SAR imagery," Canadian Journal of Remote Sensing, Vol.27, No.5, October 2001.

27. Benchmark data sets,

    `http://ida.first.fhg.de/projects/bench/benchmarks.htm`

DISTRIBUTION LIST

Detection and Classification of Objects in Synthetic Aperture Radar Imagery

Tristrom Cooke

	Number of Copies
DEFENCE ORGANISATION	
Task Sponsor	
DGICD	1 (printed)
S&T Program	
Chief Defence Scientist	1
Deputy Chief Defence Scientist Policy	1
AS Science Corporate Management	1
Director General Science Policy Development	1
Counsellor, Defence Science, London	Doc Data Sheet
Counsellor, Defence Science, Washington	Doc Data Sheet
Scientific Adviser to MRDC, Thailand	Doc Data Sheet
Scientific Adviser Joint	1
Navy Scientific Adviser	Doc Data Sheet and Dist List
Scientific Adviser, Army	Doc Data Sheet and Dist List
Air Force Scientific Adviser	Doc Data Sheet and Exec Summ
Scientific Adviser to the DMO	Doc Data Sheet and Dist List
Information Sciences Laboratory	
Chief, Intelligence, Surveillance and Reconnaisance Division	Doc Data Sheet and Dist List
Research Leader, Imagery Systems	Doc Data Sheet and Dist List
Head, Image Analysis and Exploitation	2 (printed)
Tristrom Cooke	4 (printed)
Guy Blucher	1 (printed)
Tim Payne	1 (printed)
Mark Priess	1 (printed)
David Crisp	1 (printed)
DSTO Library and Archives	
Library, Edinburgh	2 (printed)
Defence Archives	1 (printed)

Capability Development Group

Director General Maritime Development	Doc Data Sheet
Director General Capability and Plans	Doc Data Sheet
Assistant Secretary Investment Analysis	Doc Data Sheet
Director Capability Plans and Programming	Doc Data Sheet
Director General Australian Defence Simulation Office	Doc Data Sheet

Chief Information Officer Group

Head Information Capability Management Division	Doc Data Sheet
AS Information Strategy and Futures	Doc Data Sheet
Director General Information Services	Doc Data Sheet

Strategy Group

Director General Military Strategy	Doc Data Sheet
Assistant Secretary Governance and Counter-Proliferation	Doc Data Sheet

Navy

Director General Navy Capability, Performance and Plans, Navy Headquarters	Doc Data Sheet
Director General Navy Strategic Policy and Futures, Navy Headquarters	Doc Data Sheet
Deputy Director (Operations) Maritime Operational Analysis Centre, Building 89/90, Garden Island, Sydney Deputy Director (Analysis) Maritime Operational Analysis Centre, Building 89/90, Garden Island, Sydney	Doc Data Sheet and Dist List

Army

ABCA National Standardisation Officer, Land Warfare Development Sector, Puckapunyal	Doc Data Sheet (pdf format)
SO (Science), Deployable Joint Force Headquarters (DJFHQ)(L), Enoggera QLD	Doc Data Sheet
SO (Science), Land Headquarters (LHQ), Victoria Barracks, NSW	Doc Data Sheet and Exec Summ

Air Force

SO (Science), Headquarters Air Combat Group, RAAF Base, Williamtown	Doc Data Sheet and Exec Summ

Joint Operations Command

Director General Joint Operations	Doc Data Sheet
Chief of Staff Headquarters Joint Operation Command	Doc Data Sheet
Commandant, ADF Warfare Centre	Doc Data Sheet
Director General Strategic Logistics	Doc Data Sheet
COS Australian Defence College	Doc Data Sheet

Intelligence and Security Group

Assistant Secretary, Concepts, Capabilities and Resources	1
DGSTA, DIO	1 (printed)
Manager, Information Centre, DIO	1
Director Advanced Capabilities, DIGO	Doc Data Sheet

Defence Materiel Organisation

Deputy CEO, DMO	Doc Data Sheet
Head Aerospace Systems Division	Doc Data Sheet
Head Maritime Systems Division	Doc Data Sheet
Program Manager Air Warfare Destroyer	Doc Data Sheet
CDR Joint Logistics Command	Doc Data Sheet

UNIVERSITIES AND COLLEGES

Australian Defence Force Academy Library	1
Head of Aerospace and Mechanical Engineering, ADFA	1
Hargrave Library, Monash University	Doc Data Sheet

OTHER ORGANISATIONS

National Library of Australia	1
NASA (Canberra)	1

INTERNATIONAL DEFENCE INFORMATION CENTRES

US - Defense Technical Information Center	1
UK - DSTL Knowledge Services	1
Canada - Defence Research Directorate R&D Knowledge and Information Management (DRDKIM)	1
NZ - Defence Information Centre	1

ABSTRACTING AND INFORMATION ORGANISATIONS

Library, Chemical Abstracts Reference Service	1
Engineering Societies Library, US	1
Materials Information, Cambridge Scientific Abstracts, US	1
Documents Librarian, The Center for Research Libraries, US	1

SPARES

DSTO Edinburgh Library	5 (printed)

Total number of copies: printed 20, pdf 19

DEFENCE SCIENCE AND TECHNOLOGY ORGANISATION DOCUMENT CONTROL DATA				1. CAVEAT/PRIVACY MARKING
2. TITLE Detection and Classification of Objects in Synthetic Aperture Radar Imagery			3. SECURITY CLASSIFICATION Document (U) Title (U) Abstract (U)	
4. AUTHOR Tristrom Cooke			5. CORPORATE AUTHOR Defence Science and Technology Organisation PO Box 1500 Edinburgh, South Australia 5111, Australia	
6a. DSTO NUMBER DSTO–RR–0305	6b. AR NUMBER 013-591		6c. TYPE OF REPORT Research Report	7. DOCUMENT DATE February 2006
8. FILE NUMBER 2005/1075428/1	9. TASK NUMBER JTW 04/202	10. SPONSOR DGICD	11. No OF PAGES 189	12. No OF REFS 90
13. URL OF ELECTRONIC VERSION http://www.dsto.defence.gov.au/corporate/reports/DSTO–RR–0305.pdf			14. RELEASE AUTHORITY Chief, Intelligence, Surveillance and Reconnaissance Division	
15. SECONDARY RELEASE STATEMENT OF THIS DOCUMENT *Approved For Public Release* OVERSEAS ENQUIRIES OUTSIDE STATED LIMITATIONS SHOULD BE REFERRED THROUGH DOCUMENT EXCHANGE, PO BOX 1500, EDINBURGH, SOUTH AUSTRALIA 5111				
16. DELIBERATE ANNOUNCEMENT No Limitations				
17. CITATION IN OTHER DOCUMENTS No Limitations				
18. DSTO RESEARCH LIBRARY THESAURUS Synthetic aperture radar Target detection Target classification				
19. ABSTRACT This is an amalgamation of a number of reports written by the author when he was contracted to DSTO through CSSIP. The reports concern the detection of faint trails, and the theory and evaluation of a number of existing and novel methods for the detection and classification of ground and maritime targets within SAR imagery. The publication of this collection allows the results to be available within defence and the wider community.				